American Academic Scientific Research Journal for Engineering, Technology, and Sciences
ISSN (Print) 2313-4410, ISSN (Online) 2313-4402
http://asrjetsjournal.org/

Adaptation Conflicts of Heterogeneous Devices in Iot Smart-Home

Motaz Osman Ahmed[a*], Associated professor Salah Elfaki Elrofai Elfaki[b]

[a,b]*Sudan University of Science and Technology, College of Computer Science and Information Technology, SUDAN*

[a]*Email: motazosman@gmail.com*

Abstract

A promising technology such as Internet-of-Things have been introduced into traditional homes, buildings and cities to become smart and offer a wide range of services to simplify and enhance people's lifestyle, a complex rule structure with a large number of sensing and actuating devices increases the chances of creating rules with faulty behaviors. Detection of sophisticated conflicts in an IoT system is one example of such faulty systems. In this paper, a mechanism is presented to detect such sophisticated conflicts among multi-resident smart-home services. Formally a model considering the functional properties of devices to distinguish a specific new kind of conflicts among the other basic types. Service User Regularity (SUR) conflict detection algorithm is proposed to trace resident habitual usage and behaviour conflicts and regulate them within the rules of the smart-home IoT-system. The system achieved good result; it could detect a reasonable number of targeted type conflicts within a synthesized data set.

Keywords: IoT; Smart-Home; Conflict; PnP; Smart Transducer; Interoperability; Integration; Adaptation; SUR.

1. Introduction

Internet of Things (IoT) is a network interconnection of everyday objects that surround us empowers with capabilities such as sensors, actuators, and communication connected to the Internet in order to provide new services and increase the efficiency [1]. As the IoT is the emerging recent advance technology growing rabidly include Radio Frequency Identification (RFID), Wireless Sensor Networks (WSNs), mobile phones, and actuators. IoT involve in many domains in which can play a remarkable role and improve the quality of our lives such as healthcare domain, transportation domain, industrial domain, Farming & agriculture domain, smart-home, smart-building, ambient and environment domain, security and emergency domain.

---

* Corresponding author.

One of the IoT biggest challenges is the interoperability, Interoperability is the common problem in the IoT realm due to recent developments tremendously introduced several heterogeneous systems and devices, some of the open issues of the interoperability are the device adaptation, device integration and service conflict. Smart-home domain is an entity with various Internet of Things (IoT) common service functions for managing its environment [2], Smart-home environment suffer from some challenges like rule conflicts. Rules; provide control to the services function of devices, these services function could face conflicts, in general; conflicts are natural phenomenon undesirable effects resulting from the direct or indirect interactions between devices [3]. In smart-home; conflicts could occur when two or more events generated by heterogeneous systems need to be triggered at an instance of time [4]. Heterogeneous systems in smart home environment comprise of devices like, CCTV cameras, energy management units, fire alarms, and other network based devices, these are often typical smart home environment implemented under a centralized architecture along with other shareable IoT service devices like AC, TV, Stove, Heater, Fans, Lamps, Air ventilator and Sprinkler for backyard garden beside doors and windows operated by actuators. Newly added device in many domains like smart house can cause a serious conflict in many different levels. Usually in most occasions the word service-conflict and device-conflict are used interchangeably. Generally there are three strategies are used for constructing system to manage smart-house namely Rule-Based, Agent-Based and ontology-based strategy, the former one is the one witch the paper will go through. The major contribution approach is its capability to detect habitual conflicts whose rules conditions cannot be satisfied or which are in conflict with other rules. The rest of this paper is organized as follows. Section two Related Works. Section three Environmental Definitions. Section four The Proposed model for conflict Adaptation, summary and conclusion are given in Section five.

2. Related Work

The contribution of each work can be measured by two criteria, first how many conflicts can detect and second how many types of conflict can recognize, beside of detecting main types of conflict also each work has its own special effort to detect special type of conflicts and provide algorithm for their resolution, usually these types of conflicts are described as complex conflict. Normally the resolutions of the basic type of conflicts are already known and just need to be applied when detecting them. Off course other features are counted as a plus for any work like efficiency, accuracy, dynamicity and multi/single resident. One more thing, the implementation method has a great deal for detecting more complex conflicts specially when joined with semantic, context-awareness and artificial intelligent (AI). In [5] the approach has the ability to detect all types of basic conflicts, but mainly it focus in incompleteness conflicts therefore the approach introduce a mechanism for detecting incompleteness rules, incomplete rule can lead to a conflict called incompleteness conflict. Two situations can cause the incompleteness first a set of IoT rules are considered incomplete if they are do not cover all the possible sensor values. Second if a set of IoT rules don't have an anti-action for their rules.

Example:

Rule 1: if temperature is greater than 70 ◦F, turn on the air-conditioner.

Rule 2: if temperature is less than 80 ◦F, turn off the air-conditioner.

For the values between 70 ◦F and 80 ◦F, the air-conditioner cannot perform both turn on and off operations at the same time.

Another example:

Rule 1: if soil moisture is less than 20%, turn on the sprinkler. When the soil moisture is less than 20%, the system turns on the sprinkler and the soil moisture keeps increasing. At some point, the soil moisture is high enough that adding more water can flood the hall yard. To complete the rule, a "turn off" action for the soil moisture values more than 20% is needed. The introduced scheme to resolve the conflict is divided into three parts first rule is decomposition to DNF then second rule relationship is established and third finally rule conflict incompleteness is performed based on relationship. The work in [6] although it mostly detects "opposite" conflict but it has special treatment for Independent / Environmental conflict, the approach detect rule priority using a concept of Entropy and Gain information to introduce mechanism called Service Usage Habits (SUH). This type of conflict appears when the smart-home is a multi-resident. For instant, assume that a resident has a habit of turning the light "on" in the living room while watching his favourite program on TV. Assume that the other resident has the habit of turning the light "off" while watching the same program on TV. Consequently, a service conflict occurs since the light service cannot satisfy multiple residents' requirements based on their habits, for that the author propose approach for a priority conflict detection based on residents' service usage habits SUH. The model SUH is built with a consistency score for each service usage to measure dissimilarity between the residents' usage requirements according to non-functional property of the serviced device through a developed conflict detection algorithm using the temporal proximity strategy to prune insignificant and loosely correlated service usage requirements. Likewise in [4] also work in rule priority with slightly differences, while [6] work in user habit [4] works in independent conflicts presenting a rule-based conflict resolution framework using weighted-priority scheduling algorithm for managing smart home environment based on Event-Condition-Action (ECA). This weighted scheduling mechanism is known as ECA Priority Scheme (EPS), if corresponding received event is enabled with highest priority, then the event weight is queried. The work [7] has much accuracy because it uses hybrid detection algorithm for detecting "opposite conflict" more than other conflicts types. Also it has less efficiency and that because it support multi-resident, different residents may have different service requirements so more conflicts may arise, However more effort is needed to enhance the efficiency which caused by multi-resident requirements, and also lack of dynamicity. In the other hand work [8] is multi-resident also, it detects most basic types of conflicts with a great deal of efficiency but less accuracy and also support no dynamicity. What distinguishes this work is based on IFTTT rules which give it ability to detect feedback & chain conflict rules. IFTTT is a Trigger-action model stand for " If That Then That", it's the one of the most direct ways to configure system behavior by identifying triggers for instant, "if there is motion" then "turning of the lights". While many work go directly to detect and resolve conflict here the things are a little bit different the work [9] overcomes the conflicting issue when using mashup services, many mashup services created by application utilize if-then-else method and that because of its easiness, when a number of mashup services become too much, it may be suffered by service conflict, this work propose the concept of context descriptor to detect service conflict and visualize their situations, another distinction for the this method is able to detect static and dynamic conflict while finding mashup service chains. There are a few works focus on detecting conflicts in single resident smart home. This type of conflict is fundamental in understanding the

general case of conflicts in multi-resident smart homes. The work [3] is of this type; single-resident framework, generally, this setting leads to efficiency and accuracy because single-resident doesn't have much interference because of the absent of the other residents, the approach of this work has the ability to detect two more type of conflicts over the basic types, they are the Additive and Transitive environmental conflicts, these two types are contextual, contexts and conflicts are highly correlated, so the approach models different types of contexts into ontology to capture various type of context-related service. The Additive-environmental conflict happens when two services update a shared variable in the same way; their cumulative effect may exceed the comfortable threshold. When the heater in the kitchen is heating while the stove is being used to cook, the temperature may exceed the comfortable temperature threshold 26C. The Transitive-environmental conflicts happens when a service is being used, its effect may trigger another service, when it is dry in the living room, the humidifier is invoked to increase the humidity, the humidity in the room increases dramatically if forgotten to turn off which triggers the dehumidifier to work. Beside the main basic conflicts there are many other conflicts but they are inconsistency in rule conflict and most of them related to ambient and environment surrounding and need context-awareness to detect them, the work [10] automatically validate the consistency of the rules for context-awareness as well as the ability to recommend adding sensors. Actually inconsistency has two type, inconsistency in rules cause by human which is represent the challenge of rule definitions for complex systems comes from the complexity of the rules, and the Rule-based reasoning engines rely heavily on the expert user's work, and therefore are vulnerable to human errors. The other type is the ambient and environmental rules inconsistency, the model is capable of detecting inconsistent rules whose conditions cannot be satisfied or which are in conflict with other rules. Introducing two primary features the space-factory and the rules-factory to refactor the content to a finite set of templates through formal validation of the model. Yet the second primary feature the space-factory is a method to define a smart-house through more intuitive, more concise and less error-prone than writing the ontology manually, therefore minimize inconsistency in rules caused by human factor. Conflicts may occur as a result of intervenient [11]. Even a conflict occurs in a single-resident home because of having conflicting intentions (comfort vs. energy saving) [12]. A conflict may emerge between two residents when they use a resource simultaneously. There is a large body of work in conflict detections that span several areas such as telecommunication systems, software ecosystems [13], smart cities[14], smart homes [15][16][17] [13][5]. In particular, conflict detection in area of smart homes is relevant to our work. Researchers have proposed various approaches like UTEA [14], IRIS [23], SIFT [24] to develop IoT smart-home architectures. The UTEA (user, trigger, environment and action) scheme to detect rule conflicts in a smart house application. UTEA classified conflicts in 5 categories and has a bunch of rule relationships for expressing events. Unlike other methods, UTEA considers user priority in conflict detection. IRIS (Identifying Requirements Interactions using Semiformal methods), this framework detect interactions between policies in the smart-home. SIFT is an architecture which provides a safe way to configure IoT devices.

3. The Environmental Definitions

To illustrate the service-conflict adaptation, the anticipation of home dweller's intentions and subsequently taking proactive action to assist them in their daily tasks, some sort of regulation to their favourite of convenient accommodation have to be setup by a management system, these systems generally fellow three strategies to manage smart-home namely Rule-Based, Agent-Based and ontology-based strategy, most relevant works used

the former one, and so as ours. Rule-based systems use rules to govern devices for conformable services to the dwellers, these rules occasionally face conflicts with each other. Conflicts can be defined as two or more actions which cannot co-exist at the same time [5]. Rule-Based is a rule in an IoT system contains trigger conditions and actions [18], a rule "if temperature is greater than 70 ◦F, turn on the air-conditioner" condition : (if clause).

Trigger: (temperature which is greater than 70 ◦F)

Action: (turn on the air-conditioner)

The simplest trigger condition has three parameters: {sensor value, operator, threshold}, from the example above the sensor value is temperature, the operator is ">", the threshold is 70F. A trigger also could have an Anti-trigger called Anti-action which represent as "else" part. In order to overcome the conflict, first you should detected it then resolve it, the hall process is depend on how the detection algorithm is effective and accurate, since it's the main factor for automating the conflict resolution, the resolution mechanism itself is reasonably simple, once you successfully detect the conflict it could remove the rule which cause that conflict or adjust the rule or assign priority to the rule or rise acknowledge to the manager. However sometimes the resolution is not that easy, sometimes become deeply sophisticated when come across such complex rule structure; such a rules can be defined as a program using an event-based programing language. A conflict may arise based on four different kinds of sources, when multiple users compete over a single resource, an AC; it is called a resource-level conflict [21]. When various applications concur over a resource, building administration applications attempting to control a room's lighting, it is called an application-level conflict [22]. When conflict arises because of different contextual policies, it is called a policy level conflict [23], a user listens to music through his smartphone device inside a library; it violates the library's silence policy. When there are diverse user inclinations in a similar setting, one user wants to use the light keeping at its full luminosity, and another user likes to stare at the TV keeping the light at its half luminosity, a conflict may occur. It is regarded as a profile level conflict [24]. Figure 1 shows the smart-home with its component, the components can be divided into three categories of the house devices, centralized network based devices like (CCTV cameras, energy management units, fire alarms) and shareable IoT service devices like (TV, DVD, radio, AC, heater, light and fan) and non-sharable devices like(microwave oven, toaster, electric kettle, and washing machine). Sensors and actuators are essential for IoT for automating things like doors, windows and curtains. Small farm or guarding surround the house also can be a part of smart-home, smart-home could be habited by single resident or multi-residents, the more residents reside in a house the more conflict a rises because of their preference and likewise the number of devices. Sensors could be real sensors or virtual sensor, thermometer for temperature is a real sensor, weather casting for rain or windy day is a virtual sensor.

Any smart device has a properties these properties have two types, Functional & Non-Functional properties, a lamp; the functional is providing illumination the non-functional are luminosity level, durability and power consumption, TV the functional is television casting the non-functional are volume, channels, resolution and connectivity. Always the non-functional properties are responsible for the conflicts.

Figure 1: Smart home with its component

3.1 Conflict Classification (taxonomy)

Generally the taxonomy of conflicts names are classified into three main types, the names of these types are defer from paper to paper but the classification itself remain the same, in [5] classification names of conflicts are "Execution", "Shadow", "Independent", in [19] are named "opposite", "Overwrite" and "environmental" . Anyhow it doesn't matter the names while the types are remain the same. The mentioned conflicts taxonomy are called Direct-Conflict, The Indirect-Conflict is described by the same paper as "Chain" & "Feedback" conflict, both of them are almost similar, the first one is a sequence of rules could make one rule leads to another and the other invoke third action the action lead to fourth one in which chain of rules leads to a conflict, Feedback conflict indicates that there can be a loop where one sensor causes an action which effect another sensor, the another sensor causes an action affects the first sensor. One more type of conflict is the Incompleteness-Conflict, always rules exist as pair one for ON and one for OFF, formally called Action and Anti-action rule.

Rule1: if soil moisture is less than 20% turn on sprinkle

Rule2: (a rule for turning off should be here)

If there is no rule for turning off the sprinkle the yard will be flooded by water, so this is incomplete rule. Also a set of rules are considered incomplete if they are not cover all the possible sensor values.

3.2 Service Event Definition

Service event structure written in the form of tuple (list), generally a tuple could be (single, double, triple, quadruple, quintuple, sextuple ..) sometime event structure could be very complex and hard to detect and resolve, the expression syntax may vary from framework to framework depend on the planner how they formulate the polices according to their algorithm and the approach they follow. Anyhow the simplest structure may look like this;

Service event ({S_{id}, F, Q},T ,L ,U). Where;

S_{id} : Service identifier

F : Functional property

Q : non-functional property

T : Execution time also denoted as $\{T_s, T_e\}$ for start & end

L : Location

U : users

E.g. Service event ({3, {telecasting pro}, P35dB, 50 unit, ∞}},{9:30,10:30},living room, 5)

The following are the three types of conflict;

a) Execution conflict

Rule1: if time is 7pm then turn the light ON

Rule2: if nobody home then turn the light OFF

b) Shadow conflict

Rule1: if soil moisture is less than 30%, turn ON sprinkle

Rule2: if soil moisture is less than 20%, turn ON sprinkle

c) Independent conflict

Rule1: if dark outside, close the window.

The outside photo sensor affect the inside window actuator. In Execution-Conflict for one sensor value two different rule works against each other result in conflict, the resolution is to adjust the range of one rule. In Shadow-Conflict a redundant rule exist, one rule is a superset of another one, the resolution is to remove one of the rules. Whereas in Independent-Conflict it's similar to execution instead of having overlapping range it has distinct rules from distinct sensor or two sensors are locate in deferent environment, the resolution is to assign priorities.

The table shows the most essential conflicts, conflicts may arise for many reasons, many types of conflicts could be defined depend on the complexity environment of the smart-home, conflict could be contextual-conflict, semantic-conflict, occurrence-conflict (a delay from sensor), authorization-conflict, action-conflict (appliance and environment interactions). Conflicts also can be static or dynamic known as implicit and explicit service chains. Static conflict can be detected at design time. If two registered mashup services trigger contradictory actions for the same instance, then they have static conflict. However, even though a user sets mashup services

with static conflict, these services may not run at the same time. On the other hand, dynamic conflict can only be detected at runtime. Dynamic conflict can be happened because some policy conditions may depend on external input, policy actions that are triggered indirectly in an in-deterministic sequences, or missing information.

Table 1: Shows Basic Conflict Types, Each One with its Other Names

Conflicts	**Resolution**
Opposite /Execution	Adjusting the range of the rules
Shadow / Overwrite / sub-setting	Removing one of the rules
Independent / Environmental	Assign priorities
Opposite-environment	Informing the resident
Additive-environment	Informing the resident
Transitive-environment / chain / loop conflict	Assign priorities
Incompleteness	Add an Anti-action rule

3.3 Important of conflict detection and metrics

To imaging how important of conflict detection, consider the following data-set of rules which contains only 5 rules and then see how many conflict will occur.

Table 2: Shows data-set of 5 rules

Rule	Conflict range
Rule1	Soil moisture < 10%
Rule2	Raining ∧ time is 7am ∧ soil moisture Є [25,30])
Rule3	It's raining when soil moisture < 20%
Rule4	It's raining when soil moisture < 10%
Rule5	It's raining when soil moisture < 30% ∧ time is 7am

Table 3: Shows the conflicts and their resolutions

Conflicting Rules	Conflict Type	The Resolution
Rule1 & Rule5	Shadow	remove rule 5
Rule2 & Rule4	Execution	Adjust trigger range
Rule1 & Rule3	Independent	Assign priorities
Rule5 & Rule3	Independent	Assign priorities
Rule2 & Rule3	Independent	Assign priorities

These are only 5 rules, 5 conflict occur, imaging data-set with 80 or 100 interties and intended to enter a new rule, first of all its difficult to remember most existing rules, upon that the added entry may goes against many existing rules, the resulting conflict may leads to chain conflict or cause a system to enter a loop.

So the quality metrics can be determine as follow

1- Detecting all conflicts (max detection).
2- Ability to recognize most types of conflicts.
3- Ability to detect incompleteness rules.
4- Capturing conflict before it occurrence.
5- Offering resolution for complex events.
6- Comparison time of matching new rule with existing (efficiency).

The sixth one is not that much important as the numbers of rules are relevant few generally ranged between 50 to 150 rules, it would be critical for the efficiency factor if there were a thousand of rules and the system tries to capture a potential conflict before or during its occurrence.

4. The Proposed Model for Conflict Adaptation

The proposed model has the ability to detect unpredictable and random behavior which so called habit, the model approach called Service User Regularity (SUR), the model discover that the impact of routine of daily life style can be break by several factors, systematic and well organize smart-home can improve life quality and bring many benefits, but in reality not all thing go so smooth as it was planned, unexpected action can happen suddenly, up normal behavior or un wanted accidentally intentions can occur, beside that the habitual behavior can also take place, addition to that the human is a moody being, mood is temporary disorder and can be spoilt at any time and off course it's not counted when rules were set, so that the usage of devices according to the resident's mood or habit may lead to conflict. The system should be aware and has some sort of intelligent detection mechanism to automatically update or modify the rules temporary according to the context of the situation or notify the resident manager. This work can match the rivals in the sense of problem oriented approach but it has to be embedded in a full framework to get the most of the smart-home concept. The conflict detection model presented in this dissertation is based on Rule-based which guided by Event Condition Action (ECA) approach. Rule-Based is a rule in an IoT system contains trigger conditions and actions [18]. Trigger-action model is one of the most direct ways to do this. Users configure system behavior by identifying triggers, if there is "a motion" and resulting actions"turning ON the lights". Conflicts in a smart home can be due to the scale of the system, diversity in services, number of residents or variety in services-home interaction. Conflicts can be detected during the design phase of services or at runtime as not all combination of services running simultaneously can be anticipated. Some conflicts may only occur as a result of dynamic conditions or environmental changes but in contradict conflict occur as a result of a resident has yet not taken into consideration, validating the consistency of the rules upon the resident habit it's a challenging issue; If the window is closed at 7am because of the airconditioner is cooling, window is requested to be opened at 7:15 am for fresh air. There is a conflict over the window upon the resident willingness to inhale a fresh air. Opening window for refreshing and closing window to keep low temperature could not be met at the same time. To illustrate the concept of the model, most probably the work focus on functional properties of shareable IoT services where conflicts may arise. A shareable IoT service serves multiple users at the same time and location. Television, DVD, AC, light, heater, and fan are some examples of shared IoT

services. A non-shareable IoT service serves only one user at a time, toaster, microwave oven, electric kettle, and washing machine are of that type.

Notations:

Service event: a tuple of (R_{id}, device#, trigger $\{T_s,T_e\}$, action, L, U)

R_{id}: Rule identifier

Device#: Device identifier (device#, #... means same type of device)

Trigger: Trigger condition

Action: (on/off, open/close)

T: Execution time $\{T_s, T_e\}$ start, end

L: Location

U: User (resident)

Sample:

(Light)

Event ({1,{lamp1,2},{9:30, 10:00}, ON, living room,1})

(Air-conditioner)

Event ({2, AC, {1:00, 2:30}, ON, living room, 2})

(Window)

Event ({3, Window1, {3:00, 4:14}, Open, living room, 3})

4.1 Resident Dataset History (RDH)

Resident Dataset History It's a model unit, record the residents' interaction with the services. It is possible to understand different peoples service usage preferences by observing their service usage history. Different residents may have different preferences over a service usage, this unit trace the resident usage habit an behaviour for a period of time and analyse the consistency which may cause a conflict, the random redundancy for specific user may have some sort of regularity, the personality of each user differ from one to another even when come across same situation, in another word for same situation a person may behave different from another and if the situation happen again and again the same user will behave in the same way in each time, so

some kind of regular randomness is occur.

4.2 The notion of the model

From this point the role of the model Service User Regularity (SUR) conflict detector will come into picture to analyse the regular random usage of the residents habit and behavior upon their personalities and the model detect the conflict resulted from the resident against the service, related work [6] discuss the compete of dwellers of each other over a service and introduce a model called Service User Habit (SUH) conflict detector, the proposed model has gone farther to detect more complex conflict of the residents them self over services.

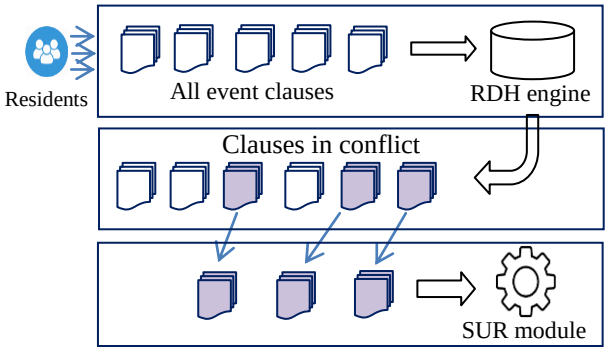

Figure 2: Model compose of SUR module and RDH engine

Figure 2 shows the component of the model, the model contains Rules Dataset (RD), Resident Dataset History (RDH) and Service User Regularity (SUR) module, the approach goes into three stages

1- Collect all events into RDH, regardless of the conflict type from all services and users.
2- Filtering out targeted conflict that's what the model is concern about.
3- SUR inspect the conflict and analysed it with the help of RD and RDH

These three stages describe the process of detecting special form of conflict, these conflicts regarding the habits and behaviour of the residents, the SUR module trace the events of all residents through a period of time and tries to anticipate the suspicious habitual conflicts, the targeted conflict is consider a conflict when an event has a sequence of intervention a temp to violate specific rule. Because this type of conflict may compose of other types the already known resolution, the resolutions may not work here, so to resolve this conflict an algorithm for dynamic adaptation module is required.

Table 5

Algorithm: SUR for conflicting events detection

Re ≡ resident
Procedure Get_Detected_conflicts(opposite,shadow,independent)
Filter out (Redundancy) *{Check history for repeated conflicts }*
For i=1 to the end of history
Count(events) when overridden _rule = i
If overridden _rule violation is more than three times the event will be in the consideration *{suspicious conflict}*
Grouping the habitual conflict and Maching with the residents
Else its normal conflict and it will be ignored
End if
Setting resolution for the habitual conflict // *Analyzation*
If clause *i* ⊂ Re *then {check the sequence of the event}*
If redundant conflict for the same Re ≈ regular then
Apply resolution count(event(median))
Else
Trigger alert *{no action}*
End if
end

Table 4: The performance comparison

Paper	**Accuracy (Threshold in MINs)**	**No. of Services**	**Conflicts Detected**	**No. of residents**
1) Conflict Detection in Rule-Based IoT Systems 2019 IEEE	**80% at threshold 2**	**95**	**27**	**5**
2)Fine-grained Conflict Detection of IoT Services 2020 (IEEE)	**83% at threshold 1**	**5000 (History)**	**200 of 230 (Overlapping)**	**4**
3)A Conflict Detection Framework for IoT Services in Multi-resident Smart Homes 2020 (arXiv)	**95% at threshold 3**	**1000 (History)**	**53 of 235 (Overlapping)**	**4**
4)Conflict Detection of IoT Services in Smart Home 2020 arXiv	**87% at threshold 5**	**60**	**15**	**1**
The proposed approach	**No threshold**	**138 (History)**	**14**	**2**

Table 4 shows a quick comparison between the most relative works, these works capture deferent types of conflicts, as we know, the performance of any approach for managing smart-home measured by many things but the most important metric is how many conflict can detect and how many types is able to recognize, from the table notice that the work 3 and 4 are depending on the history of the resident usage for detecting conflicts, that means they inspecting special type of conflict, that is clear since its shows a high degree of overlapping indicator. The threshold factor imply the accuracy of detection and how much intervention can be ignored to reach tight acceptance of conflict resolution, toning the threshold between 1 to 5 minutes is common, that depends on the amount of the sensitivity for applying resolution for some type of conflict.

Figure 3: Number of services and conflicts detected

Figure 3 explains the relationship between the number of services and the number of detected conflicts, the more services provided the more conflict may arise. It may notice that some contradicts when comparing two work together, the work 4 and the proposed approach, it notice that the proposed approach can detect more conflict as the number of services which is greater than the rival, that is true but it has to put in mine that two work are not working from the same dataset and they are in different situation. The proposed approach, process the detection over 138 interties of a synthesized data history events on a smart-home occupied by two residents, because the model is meant to detect new special form of conflict that resulted from reaction of resident against the smart-home the threshold is ignored for the system to consider the conflict as overlapping, the system obtained good result as the approach is able to catch up to 14 conflicts out of 25 suspicious state of conflict, the test bed ran on a system contains 36 rule.

4.3 Result

A test of experiment is conducted on a synthesized dataset to validate the effectiveness of the proposed approach for detecting a new form of conflicts; It's noteworthy to mention that the proposed model is able to detect unpredictable behavior conflict in terms of human usage. Compared to related works that do not consider a similar form of conflict detection the introduced approach is considered to be a nice contribution. However although the model is in the preliminary stage with minor error of detection, but it work just fine as its meant to be, and it will get all out of it when embedded in a full smart-home platform.

5. Conclusion

It is challenging to detect conflicts resulted by the behaviour of the dwellers, complex context of IoT services conflicts has various aspects and changes over time. Conflicts have different types and happened under diverse circumstances. To address this challenge, a proposed model so called Service User Regularity (SUR) conflict detector, is introduced to detect especial new type of conflict based on resident aspects and usage of the smart-home. Future work will include more advanced detection mechanism utilize context-aware and some sort of AI for adequate detection, also the advanced detection mechanism will be joined with a dynamic adaptation resolution mechanism for resolving this type of sophisticated conflicts.

References

[1]. S. H. Shah and I. Yaqoob, "A survey: Internet of Things (IOT) technologies, applications and challenges," 2016 4th IEEE Int. Conf. Smart Energy Grid Eng. SEGE 2016, vol. i, pp. 381–385, 2016.

[2]. T. Perumal, A. R. Ramli, and C. Y. Leong, "Interoperability framework for smart home systems," IEEE Trans. Consum. Electron., vol. 57, no. 4, pp. 1607–1611, 2011.

[3]. B. Huang, A. Bouguettaya, and S. Mistry, "Conflict Detection of IoT Services in Smart Home," arXiv, 2020.

[4]. T. Perumal, M. N. Sulaiman, S. K. Datta, T. Ramachandran, and C. Y. Leong, "Rule-based conflict resolution framework for Internet of Things device management in smart home environment," 2016 IEEE 5th Glob. Conf. Consum. Electron. GCCE 2016, pp. 1–2, 2016.

[5]. T. Shah, S. Venkatesan, T. Ngo, Pratima, and K. Neelamegam, "Conflict Detection in Rule Based IoT Systems," 2019 IEEE 10th Annu. Inf. Technol. Electron. Mob. Commun. Conf. IEMCON 2019, pp. 276–284, 2019.

[6]. D. Chaki and A. Bouguettaya, "Fine-grained Conflict Detection of IoT Services," arXiv, pp. 321–328, 2020.

[7]. D. Chaki, A. Bouguettaya, and S. Mistry, "A Conflict Detection Framework for IoT Services in Multi-resident Smart Homes," arXiv, 2020.

[8]. D. C. among A. D. in S. Homes, "Detecting Conflicts among Autonomous Devices in Smart Homes," 2019.

[9]. H. Oh, S. Ahn, and J. K. Choi, "Mashup Service Conflict Detection and Visualization Method for Internet of Things," no. Gcce, pp. 4–5, 2017.

[10]. H. Aloulou, R. Endelin, M. Mokhtari, B. Abdulrazak, F. Kaddachi, and J. Bellmunt, "Detecting Inconsistencies in Rule-Based Reasoning for Ambient Intelligence," Proc. IEEE Int. Conf. Eng. Complex Comput. Syst. ICECCS, vol. 0, pp. 235–240, 2016.

[11]. M. Yagita, F. Ishikawa, and S. Honiden, "An Application Conflict Detection and Resolution System for Smart Homes," Proc. - Int. Work. Softw. Eng. Smart Cyber-Physical Syst. SEsCPS 2015, pp. 33–39, 2015.

[12]. A. Abusafia, A. Bouguettaya, and S. Mistry, "Incentive-based selection and composition of IoT energy services," Proc. - 2020 IEEE 13th Int. Conf. Serv. Comput. SCC 2020, pp. 304–311, 2020.

[13]. A. S. Alfakeeh and A. H. Al-Bayatti, "Feature Interactions Detection and Resolution in Smart Homes Systems," Int. J. Electron. Electr. Eng., vol. 4, no. 1, pp. 66–73, 2016.

[14]. M. Ma, S. M. Preum, and J. A. Stankovic, "CityGuard: A watchdog for safety-aware conflict detection in smart cities," Proc. - 2017 IEEE/ACM 2nd Int. Conf. Internet-of-Things Des. Implementation, IoTDI 2017 (part CPS Week), pp. 259–270, 2017.

[15]. Y. Sun, X. Wang, H. Luo, and X. Li, "Conflict detection scheme based on formal rule model for smart building systems," IEEE Trans. Human-Machine Syst., vol. 45, no. 2, pp. 215–227, 2015.

[16]. M. Nakamura, H. Igaki, and K. I. Matsumoto, "Feature interactions in integrated services of networked home appliances - An object-oriented approach," Featur. Interact. Telecommun. Softw. Syst. VIII, pp. 236–251, 2005.

[17]. H. Hu et al., “Semantic Web-based policy interaction detection method with rules in smart home for detecting interactions among user policies,” IET Commun., vol. 5, no. 17, pp. 2451–2460, 2011.

[18]. T. Perumal, M. N. Sulaiman, and C. Y. Leong, “ECA-based interoperability framework for intelligent building,” Autom. Constr., vol. 31, pp. 274–280, 2013.

[19]. K. Kesehatan, “No TitleΕΛΕΝΗ,” Αγαη, vol. 8, no. 5, p. 55, 2019.