RESEARCH ARTICLE OPEN ACCESS

Review paper of existing Dynamic Adaptation Techniques of Heterogeneous Devices in IoT Systems

Motaz Osman Ahmed [1], Salah Elfaki Elrofai Elfaki [2]

[1] College of Computer Science and Information Technology,Sudan University of Science and Technology

[2] Sudan University of Science and Technology, College of engineering, Electronic school - Sudan

ABSTRACT
Internet of thing (IoT) consist of several devices connected together to provide various services to end user. In a typical IoT network, devices may need to be connected to the network without a prior setup, the review introduce a classification contain four categories, two categories have been covered namely Device-Conflict & Service Conflict, further more Device – Conflict divided into two sub categories namely *Defined* and *Undefined* devices. In the former, devices are predefined and can be integrated seamlessly with the network. In the later, devices need to be integrated using different techniques without human intervention. This study attempt to classify and survey the categories of the devices above and the various techniques used for each category.
Keywords - IoT, PnP, Smart Transducer, Smart-Home, Interoperability, Integration, Adaptation, Conflict.

I. INTRODUCTION

Interoperability is the key challenge in iot due to the large and heterogeneous nature of the network devices, interoperability can be defined as the ability to exchange usable data between systems and devices[1]. There are many types of interoperability such as user interoperability, device interoperability and interoperability of data. One of the open issues of the interoperability is the device adaptation, device adaptation is simply means how it can be deal and interact with heterogeneous devices connected to the IoT environments by allowing them to be added dynamically during runtime that means to achieve dynamic, automatic and plug and play integration.

The challenge of the adaptation represent in those devices facing difficulties to integrate into the network, which described as undefined devices, they should be uniformly discoverable and impeded into different platforms easily [2]. Resolution of conflicts of newly added device in many domains like smart house can cause a serious conflict in many different levels such as service, resource, authorization level. While the main cause of the challenge is that many vendors do not provide open interoperable frameworks for their propriety devices. The existing adaptation technologies lack sufficient flexibility to adapt changes, as the techniques used for integration are both static and sensitive to new or changing device implementations [2]. Most of the solutions demonstrate how all the technologies are already available but what is required is to make the right recipe of their own envision.

This paper attempts to survey the state-of-the-art in the area of IoT device adaptation. However there are several published review papers that cover different aspects. [3][4] These two review papers cover the dynamic adaptation in the sense of conflict detection in smart home domain which allow the residents to manage devices efficiency and easily, [5] Cover privacy and security in the same domain. [6] Present the enabling technologies that help in producing applications that support intelligent decision making. [7] Covers the main communication with emphasis on Radio-frequency Identification (RFID) and its potential applications. In [8] the authors address the IoT architecture and the challenges of developing and deploying IoT applications. [9] Enable technologies and application services using a centralized cloud vision. While the authors in [10] provide a survey regarding clinical applications wireless devices.

This survey spot a light on smart home domain along with some recent revolutionary standardization solutions such as IEEE 1451 Smart Transducers and Universal Plug and Play (UPnP) both harmonize and manage interoperability in transducer (sensors & actuators) networks. Transducers seemed to be the best solution to integrate a large variety of sensors and actuators only and only if vender's fellow the correlated standard, these type of devices have the ability to configure and manage themselves in device collaboration, understanding high-level information such as user behaviours and intentions.

The rest of this paper is organized as follows. Section 2 Proposed Classifications For Device Adaptation. Section 3 Device-Conflict Techniques. Section 4 Service-Conflict Techniques. Summary and Future Research are given in Section 5.

II. PROPOSED CLASSIFICATION FOR DYNAMIC ADAPTATION

The survey illustrates the Dynamic Adaptation Techniques with a classification and mainly divided into four categories Device conflict, Service conflict, Resource conflict and finally Authorization conflict. Then each of device & service conflict have two subclasses namely Rule-Based & Agent-Based for service conflict and Defined & Undefined for Devices conflict. The review will go to discuss the first two classifications in detail.

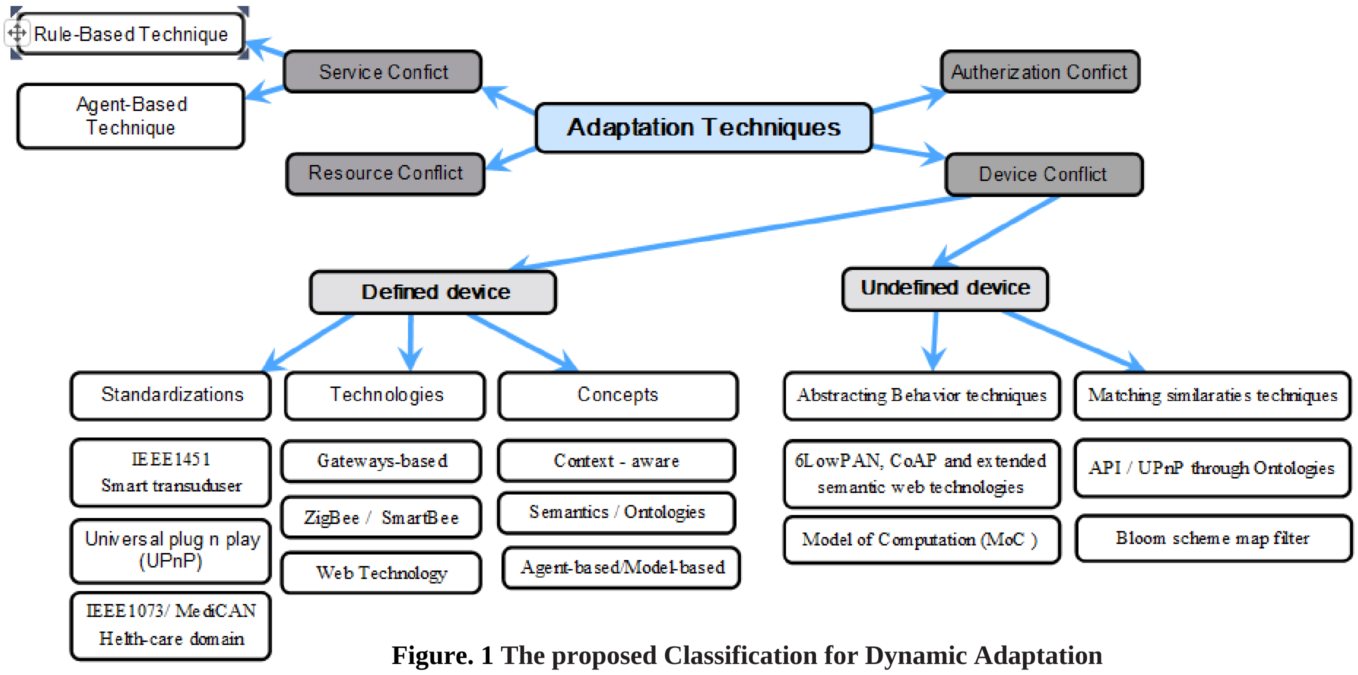

Figure. 1 The proposed Classification for Dynamic Adaptation

III. DEVICE-CONFLICT TECHNIQUES

DEFINED DEVICE ADAPTATION TECHNIQUES

The following sub-suctions show techniques use one or more base-concepts that used with defined device and the problem solved in deferent ways.

A. Concepts-based techniques

1) **Context-Aware:** Context aware technique [11][12][13][14] can play an important role in dynamic device adaptation, since they enable applications to limit the amount of information sources according to the user's context constraints.

The work In [12] two systems were built for sensing Indoor Air Quality (IAQ) the first one rely on Android smart phone based context aware compatible sensing system called Sensor drone, and the other one using Arduino based setup, the two approach has been built to measure the IAQ ventilation rate of CO2 levels in a controlled environment classrooms, the test ran in experiment obtained good result. However: measuring IAQ in different types of rooms is needed for better adequate readings.

Also paper [11] Technique's based on context-aware merging with Simple Network Management Protocol (SNMP) protocol result in Intelligent Transportation Systems (ITS) as extendable SNMP agent, enabling dynamic adaptability using Fuzzy Cognitive Map logic.

The architecture's resemble a model called Split Cycle and Offset Optimization Technique (SCOOT) for controlling traffic light data, it characterized by readability, modularity, extensibility and low latency and low code complexity.

The work in [13] presents the integration of 'Self-configuration with Wireless Sensor Networks (WSN), based on Received Signal Strength Indicator (RSSI), the author experiment Lighting control systems with sensors utilizing context awareness. Most Ambient intelligence (AmI) systems based on context aware concept, as in [14] enable sensors to co-exist within the environment enabling the system to recognize any situations of a technological living environment.

2) **Semantic-based Technique:** Semantic concept is usually used to reveal the meaning of the data and information generally is used with a combination of other techniques. [15] introduced a middleware solution using a common semantic with multiprotocol broker. [16] Created model provides a dynamic set of capability through semantic concepts and ontology. [2] Matches different Application programing Interface (APIs) data semantically in a

meaningful way by using Generic API Ontologies built through OWL. [17] Achieving self-configuration with the help of semantic web. In [18] the model avoid incoherent ontology during the process of calculation between the correspondences ontologies since a new type can have common semantics. [19][20] The first one utilize Semantic Gateway as Service (SGS) and distributed Semantic Sensor Web (SSW) architecture, the second one proposes a Semantic Web gateway architecture located between physical level sensors and cloud-based services. [21] Provides access to multiple heterogeneous devices' data based on semantic web and linked data principles. [22] Collecting and publishing sensor data semantically residing in MySQL database.

3) **Ontologies-Based Techniques:** The ontology technique widely known for using to resolve the interoperability and essential for mapping and monitoring, it's very complicated and time consuming, dynamic control interoperability needs monitoring the dynamic events and dynamically generating or reconfiguring the capability workflows, for that the author of [16] introduced a middleware solution regarding this problem, the model called InteropAdapt, by providing a dynamic set of capability workflows and handling through concepts of ontology, the model address's the key problem of dynamism which are:

- Deployment of applications with new requirements.
- Changing capabilities of devices over their life cycle.
- Mobility of the device.

4) **Model-based Embedded System Technique*:*** Cyber- Physical System PnP technology (CPSs). One of the most popular complex IoT-based solution technologies employed for plug and play transducers, unlike IEEE 1451 which is a high level byte oriented. The proposed solution in [23] complement IoT with additional low level bit oriented functionalities and independent of the transducer type.

5) **Agent-based Techniques:** Agent-based defined as being networked software programs that can perform specific tasks for a user and possessing a degree of intelligence that permits them to perform parts of their tasks autonomously and to interact with their environment in a useful manner [24]. Agent-based incorporating PnP becoming the common solution to integration issue for medical devices. The work in [25] implements agent-base in portability small device (PDA, smart devices), the frame work stand on the concept of Role system. However the new device connecting to the system must be equipped with a vendor-specific role. Although multi agent-based applications are an appropriate way to cope with the growth and evolution of IoT systems a few results about agent-based architecture emerge the IoT ecosystem as in [26][27][28].

B. Technology-based techniques

1) **Gateways:** Gateways, although they are devices but they are also considers as a method for achieving interoperability, they are the most complex internetworking devices, sometimes called network connectors and Protocol Converters, they acts as a center of data communication between the physical world and the Cloud. The data is semantically annotated at the gateway and hence these services can exploit the sensor information for further analysis. [19] Got architecture based on Semantic Gateway as Service (SGS), [20] Proposes a Semantic Web gateway architecture located between physical level sensors and cloud-based services.

Also there is another kind of gateways called Multi-Technology Smartphone As Gateway, likewise, some work a tempt based on a smartphone devices as in [29] because it contains all technologies that help to remove the barrier of the interoperability, it has multi-standard, multi-interface and multi-technology communication and radio interfaces. The work proposed high-level software architecture for discovery/management and data collection, processing and forwarding to Internet and Cloud using.

2) **SmartBee*:*** SmartBee as gateway mainly used for interaction and monitoring. [30] Introduces SmartBee Gateway solution for home or office automation bridges the gap between an IP and the ZigBee controlling or monitoring end devices, the work mainly addresses the issue of integration of multiple vendor specific devices into a single smart solution, so the venders don't bother them self about the gateway any more but only on their devices, moreover users can control or monitor these devices through internet using web interface or through their mobile phones. However it requires the manufacturer to produce their own type of SmartBee enabled end devices by following the SmartBee Specification.

3) **ZigBee*:*** Is an IEEE 802.15.4 designed for high-level communication protocols, suitable for small scale projects network such as home automation, medical device specified by low-power, low data

rate. [31] Present a method for making wireless body-worn medical sensors aware of the persons which they belong, to enabling any non-technical user to form a wireless network plug and play by just sticking devices to the body, the challenge here If there are no physical wires connecting the sensors into the network, then how do sensors know to which person they belong to? As a consequence a radio cannot distinguish whether a sensor belongs to the same or another person, anyhow the idea is to use body-coupled communication (BCC) using the human body as signal transmission medium. However, the sensitivity of the system must be increase to place medical sensors all over the user's body and also to setting up such a wireless network is still a difficult task.

4) **Web Technology:** Integrating IoT devices with the World Wide Web (WWW) has taken consideration by [32] which introduces a new approach 'PnP Web Tag', which provides web developers with an easy and systematic way of including sensor data and actuator controls in their web pages. The majority of approaches using this technique focus on an application layer network protocol Constrained Application Protocol(COAP) which map IoT resources, utilize URLs via standard set of REST operations (GET, PUT, POST, DELETE) on top of User Datagram Protocol (UDP) rather than Transmission Control Protocol(TCP), COAP. However the fully representation of COAP resources inclusion in standard web is very little, the author of [25] addressed this shortcoming by extending the HTML tag with new tag <pnp> which allows web developers to embed live IoT data in their pages.

The author of [15] propose a new kind of broker, named *QEST* that can bridge the two worlds, represented by their protocols Message Queue Telemetry Transport (MQTT) and Representational State Transfer (REST). *QEST* is a multiprotocol broker with a common semantics. However the idea is in the development state, it lacks peer-to-peer capabilities and also need to extend the *QEST* broker in order to support sensor networks and to fully support the MQTT.

C. Standardization-Based techniques

1) **IEEE1451 Smart transducer standard:** It's a set of smart transducer standard to provide a unified way for defining data exchange format and communication protocols to access any type, manufacturer, and wired or wireless sensors and actuators**,** The objective of the IEEE 1451 standard is to make it easier for transducer manufacturers to develop smart devices. A series of the standard has been developed up to version 7. The standard consists of two main components: Network Capable Application Processor (NCAP) and Transducer Interface Module (TIM). TIM contains Transducer Electronic Data Sheet (TEDS) for data format. However IEEE 1451 is a complex standard, consist of algorithms require large amounts of memory and high computing capabilities.

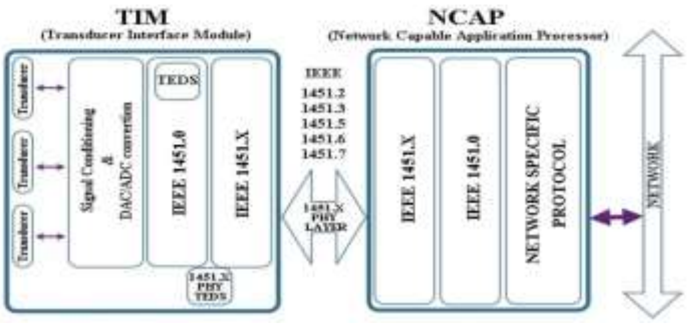
TIM
(Transducer Interface Module)
NCAP
(Network Capable Application Processor)
Transducer
Transducer
Transducer
Signal Conditioning & DAC/ADC conversion
TEDS
IEEE 1451.0
IEEE 1451.X
1451.X PHY TEDS
IEEE 1451.2 1451.3 1451.5 1451.6 1451.7
1451.X PHY LAYER
IEEE 1451.X
IEEE 1451.0
NETWORK SPECIFIC PROTOCOL
NETWORK

Figure. 2 IEEE 1451 Architecture diagram

A practical example of 1451 is in [33] which introduces a platform named Smart Transducer Integrator (STI) that allows integration of various transducers types and support different basic communications interfaces.

2) **Universal Plug and Play (UPnP):** It's a technology generally use for the computer systems. It's a distributed open networking architecture protocol that defines architecture for service discovery and sharing between networked devices. The Work in [34] integrates UPnP with web technology for more capabilities. Many work aimed to integrate UPnP with IEEE 1451 environment, such integration have been carried by [35][36][37][38]. Generally IEEE 1451 and UPnP are different in data format, command format, description and device discovery. TIM cannot run UPnP protocol because it has limited capabilities, processing, memory and power, it can only run smart sensor based on WiFi or 6LoWPAN, the TEDS for manufacturer-related information also allows self-recognition describing the identification, the NCAP responsible of all type of conversion between IEEE 1451 model and UPnP model.

The work [35] [36] presents network interface to UPnP based on 1451 shares almost the same steps, while [37] introduces approach on establishing relationships between the basic functions of smart sensors and the UPnP phases. Although this work establishes a good basis for interoperability, many open questions remain unsolved, e.g., how transducers' reconfigurations are managed and how the IEEE 1451 data streaming model fits into the UPnP event model.

It's also found that such integration in [38] which describes a new UPnP device called Sensor Manager

(SM) filling the gap between a UPnP network and IEEE 1451 transducer networks by acting as a mapping gateway.

3) **Health-Care domain:** This segment of medical domain devices exhibit wide heterogeneity and are often strictly related to one another. While health care offers a wide range of solutions in patient Tele-monitoring, nevertheless, these solutions are unpractical without the use of standardization.

The work In [39] has a contribution using IEEE 1073. However; although the model achieved the concept of PnP but the standards were originally designed for bedside environments. The Medical Information Bus (MIB) device architecture is more static, relying on resources usually resident on desktop.

The standardization committees are working towards adapting X73, the X73 where designed to address Intensive Care Unit (ICU), [40] introduce platform based on X73 for P&P MDs Medical devices (ECG, Pulse Oximeter, Spirometer, Sphygmomanometer, blood pressure, spirometry) envisioning a new profile for Personal Health Devices (PHD) communications within PAN (Personal Area Network). However MDs is strongly conditioned to the vendors, it is also difficult to find MDs with an X73-compatible physical output and manufacturers.

From another angle, wireless body area networks (WBANs) which includes Personal Area Networks (PAN) and Body Area Network (BAN) Near-field intra-body communications are emerging as wireless communications advance, enabling to attach sensors to the body for the purpose of health monitoring.

MediCAN™ technology [41] is becoming a candidate to be an open standard point-of care device communications. Its addresses communication services and protocol definitions based on Control Area Network (CAN), MediCAN also has BCC primary node is a hub connecting to a site, and DCC secondary node connects each medical instrument to the communication network via BCC. Each device has a DCC and a BCC connecting to a gateway.

UNDEFINED DEVICE ADAPTATION TECHNIQUES

In this study, contributions of this category share the method of observing devices with similar metadata, TEDs, INF files and APIs library, tuning them for encapsulation with the one from the system template, or searching for similarities among other devices and associate it to the undefined device in order to achieve the goal of PnP. The following work solutions have been divided in to two groups Abstracting Behaviour Techniques and Matching Similarities Techniques, table3 at the end of this review summarize the techniques used for undefined device .

A. Abstracting Behavior Techniques

The author of [17] came up with the means to process data directly on the devices, eradicating the need of gateways or conversion services, achieving this by introducing an approach that make true self-configuration concentrate on 6LowPAN, CoAP and semantic web technologies which are protocols available for embedded devices, using Resource Description Format (RDF), SPARQL (the standard query language for RDF) and Linked Open Data (LOD) cloud for linking data to cloud and finally SPITFIRE for adding smart self-annotation to the newly added device using fuzzy logic methods for maximum compatibility. However the author stand on semantic data and the smart self-annotation of the device presenting algorithm by which newly installed device can encapsulate their behaviour with that of other devices in the network.

Whereas in [42] the architecture used based on Model of Computation (MoC) technique, the hall idea of MoC is to encapsulate sensor and actuator drivers with abstract behaviour that provided by MoC driver, generally MoC may have MoC drivers for different MoCs like dataflow MoC, synchronous MoC and event-driven MoC. In this architecture the MoC is used for Dataflow Process Network (DPN), DPN MoC driver is generated from standard template as a C function which resides on top of the device driver and is called each time, the OS Abstraction Layer (OSAL) provides the abstraction which isolates embedded software from the real-time operating system.

While in [43] the author extends the idea from the Moc technique introducing a concept called Driver Engine Framework that overcome the problem of standardization and propriety of drivers toward specific operating system, the framework generate driver and standardize them for the device automatically from model-based templates, which has the ability to override the device driver with abstract behaviour given by the model-system using ADL language. However, the model able to maintain time-to-market by reducing faulty error (conflicts) and modelling discrepancies.

B. Matching Similarities Techniques

Ontology with APIs is the one of the state of the art solution technique which used by [2] resolving device integration, the paper proposes approach called Dynamic Data Acquisition API which allowing the heterogeneous devices to be added dynamically into IoT environments during runtime, the approach handle the issue by identifying both known and unknown devices during runtime and gathering data by identifying the devices' software and hardware specifications and APIs. This process is done after classification mechanism according to already defined APIs devices' specifications matching the known with the unknown devices assuming that the devices with the same specifications will have the same APIs, finally the devices' APIs are being illustrated into ontologies, translated into a common format, diverse APIs mapped to match the different APIs data methods semantically using Generic API Ontologies built through OWL.

RFID, although it's an old technology but it's considered one of the intrinsic element for the IoT that connects physical objects, in [44] introduce a scheme using a Bloom filter that dramatically decrease the data transmission during the identification process. The author has shown many methods scheme with their drawback of distinguishing unknown tags from known ones, the simplest one is that just collect the Identification (IDs) of known tags and unknown and then make comparison, this process will lead to massive time consuming. All methods fall into two schemes: Aloha-based and tree-based schemes.

The key problem of the author's approach is how to separate unknown tags from known tags efficiently, since every tag cannot determine itself as known or unknown. Anyhow the approach deactivate the known tags and collecting unknown tags, this action will filter known tags to prevent signal interference from known tags, then the approach collect IDs of unknown tags using an indicator vector with avoidance of tag-tag collision and multiple hash function to map elements into multiple.

The tag-tag collision itself is a problem and it's not a voidable due to the limited computation and communication capabilities of RFID tags, at the beginning, the Bloom scheme map all known tags to construct the filter, then each of all remaining unlabelled tags checks whether it belongs to this filter or not. If not, the scheme determines it as unknown and labels it. Otherwise, it labelled as known tag. However there might be some false positive errors which are conflict happens when unknown tags appeared as known.

Here is another technique uses Ontologies Integrated with UPnP, mainly for defining description to a Non-UPnP device. [45] Describes the designing applications to support multiple protocols is time consuming since developers must implement the interaction with each device profile. Several protocols coexist in smart homes but interactions between devices cannot be put into action unless devices are supporting the same protocol.

So the basic operation of the system is to deal with non-UPnP devices, which happened in the sixth stage of the system, setup automatically generates proxies using the validated ontology alignments which represent the transformation rules to go from an UPnP standard device description to a non-UPnP device description. This step is based on the Model Driven Engineering (MDE).

similar action called Base Driver in [46], there is no unified description UPnP for the same device type. To solve this problem, while annotating the description manually can be difficult and error prone the system automatically generated independent ontology from the device description, then correspondences between ontologies are calculated semi-automatically using alignment techniques avoiding incoherent ontology.

To accomplish this job the author proposes Dynamic Ontology-based proxy Generator (DOXEN) installed along with predefined templates. However If the non-UPnP device does not have an equivalent UPnP device, then, there would be no adaptation with such device type, also the alignment techniques are semi-automatic and are based on the syntax, the semantics and the structure that means a human intervention is needed to validate the detected correspondences.

IV. SERVICE-CONFLICT

To illustrate the service-conflict adaptation which is concentrate the steady on the Smart-Home domain, service-conflict can be defined as two or more actions which cannot co-exist at the same time [47], sometimes in most papers the word service and the word device are used interchangeably. Generally there are two strategies are used to resolve the service conflict namely Rule-Based and Agent-Based strategy, the former one will follow us to the rest of this section.

Rule-Based is a rule in an IoT system contains trigger conditions and actions [48]. For example, 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 is70F. A trigger also could have an Anti-trigger also called Anti-action which represent as "else" part"

In order to overcome the conflict, first it 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 it 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.

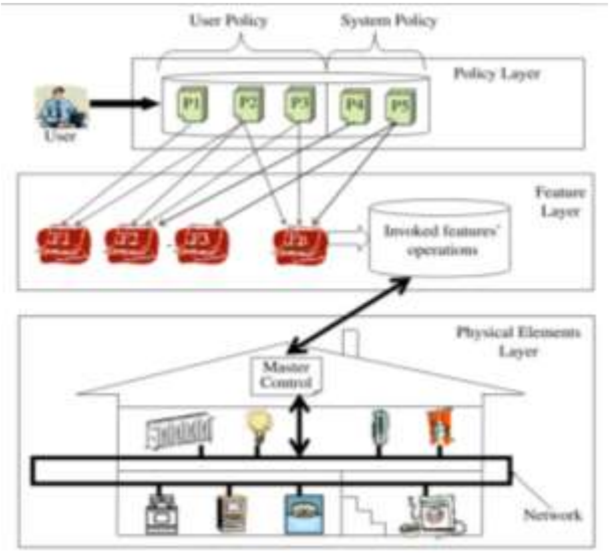

Figure 3: Smart home Features, policies, and physical elements

Figure 3 shows the smart-home with its component, the components are the house devices, sensors and actuators. 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, for example thermometer for temperature is areal 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, for example a lamp the functional is providing illumination the non-functional are luminosity level, durability and power consumption. Another example 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

Conflict Classification (taxonomy):
Generally the taxonomy of conflicts are classified into three main types, the names of these types are defer from paper to paper but the classification itself remain the same for example in [47] classification names are “Execution”, “Shadow”, “Independent”, in [49] are named “opposite”, “Overwrite” and “environmental” . Anyhow it doesn’t matter if it’s this or that, the study shall go for the former one for the rest of this section. The previous 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 other 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, for example

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. A set of IoT rules are considered incomplete if they are do not cover all the possible sensor values.

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 examples for the three types of conflict;
Execution conflict
Rule1: if time is 7pm then turn the light ON
Rule2: if nobody home then turn the light OFF

Shadow conflict
Rule1: if soil moisture is less than 30%, turn ON sprinkle
Rule2: if soil moisture is less than 20%, turn ON sprinkle

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.

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 how many conflict occur.

Table. 1 Shows data-set of 5 rules

Rule	Conflict range
Rule1	Soil moisture < 10%
Rule2	Raining ∧ time is 7am ∧ soil moisture € [25,30])
Rule3	Its 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. 2 Shows the conflicts and their resolutionsd

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 detect incompleteness rules.
3- Ability to detect all types of conflicts.
4- Capturing conflict before it occurrence.
5- Comparison time of matching new rule with existing.
6- Resolution for complex events.

Some descent platform introduced by big association for smart-home management like Amazon AWS IoT[50], AT&T M2X, IBM IoT[51], and SMSUNG Smart-Things[52] are customizable by developers support drag and drop utilizing User Interface (UI), normal user can configure his smart home easily.

Researchers have proposed various approaches like UTEA[53], IRIS[54], SIFT[55] 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 an architecture which provides a safe way to configure IoT devices.

in [56][57] both visualize ECA (event-condition-action) is a popular mechanism to execute a rule in an IoT system. In an ECA based rule architecture, events change the state of the IoT system and if this change triggers the condition, the actuator performs needed actions. [58][59][54] Petri-Net, SPIDER and IRIS approaches are consider the rule as a policy defined by a user using policy management and also detect redundant, conflicting and incomplete boolean expression.

In [47] shows a scheme to detect incomplete rules, the rule is divided into multiple sub-rules, then applying DNF (disjunctive normal form) optimization tree method for converting any boolean function to a DNF form using the distributed law property for each action a variant of the covering polygon to detect the incomplete rule.

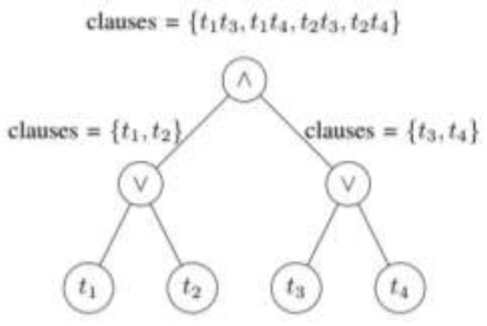

Figure.4 An expression tree for ($t1$ _ $t2$) ^ ($t3$ _ $t4$)

The concept of information-entropy and information-gain from information theory to measure the dissimilarity [60] used algorithm based on temporal proximity for a-priori conflict detection based on residents' service usage habits to detect conflicts in multi-resident smart-house. The gain scores are calculated based on different functional and non-functional requirements of IoT services. To a achieve the proposed idea the author introduce three module, Service Event Sequences (SES) to record interactions, Fuzzy Service Attribute (FSA), is a tuple set of pairs representing all possible values of the attribute with their consistency scores and Service Usage Habit (SUH).

In [56] proposed a framework is to provide an efficient conflict detection and resolution using scheduling algorithm. The rule-based conflict resolution framework is implemented using a home server integrated with Web Services, five different heterogeneous IoT systems with a conflict resolution module is based on Event-Condition-Action (ECA) rules, events generated by heterogeneous systems in smart home environment are synchronized and updated via Conflict Resolution module and screened based on their predefined priority. The Conflict Resolution module is implemented based on weighted scheduling mechanism. This weighted scheduling mechanism is known as ECA Priority Scheme (EPS). Using EPS, if corresponding received event is enabled with highest priority, then the event weight is queried.

The advantage of the developed framework is to have greater control over events generated by bespoke devices in smart home environment setting.

Another method to detect service conflict and visualize their situations is using concept of context descriptor create mashup services with if-then-else [57], many Internet of Things applications provide this method due to its simplicity. However When a number of mashup services are increased, it may be suffered by service conflict performing opposite actions or abnormal behaviour at the same time. Context is any information that can be used to characterize the situation of an entity. Each IoT instance and mashup service can change context by its action. The changed context affects conditions, so it is important to track how actions can changes context. Context descriptor describes the context changed by mashup service instances. Context descriptor consists of the following attributes; Instance, Context and Direction for changing direction of context.

The proposed method is able to detect static and dynamic conflict while finding implicit and explicit mashup service chains.

Typical structural errors in a rule based system are redundancy, inconsistency, incompleteness and circularity. [58] Addressed types of structural errors in an expert system and proposed Petri-nets formalism for verifying these structural errors. The approach can automatically detect types and causes of errors, the method result can be directly applied to existing Prolog programs for rule verification.

V. CONCLUSION

From the survey it has conducted and conclude that until now there is no "one-size-fits-all" solution, but the steady presents the envision of the current solutions. The review attempted to cover some IoT domains to resemble the rest of others, Smart-home, Health-Care, Smart-Farm all for the sake of device adaptation, some work demonstrate faulty error conflict due to addition of new devices and other experience unavoidable malfunctioning due to the nature of the work, also it did not come across work that addresses both defined and undefined devices together. The steady also made classification to covers wide verity of standardization along with many introduced techniques to elaborate the process of the device adaptation. However; although huge work were participate in the matter of IoT interoperability, but eventually the standardization will remain the main solution for the massive heterogeneity of devices over the IoT network.

Undefined Device Techniques

Paper name	Problem	Technique	Solving Method	Drawback
Plug'n'play IoT Devices An Approach for Dynamic Data Acquisition from Unknown Heterogeneous Devices, 2018 (Springer)	Integrating new devices during runtime addressing flexibility, static and sensitivity	Generic API Ontologies built through ontology web language (OWL) approach named *Dynamic Data Acquisition API*	the approach matches the known device specification with the unknown devices assuming that the devices with the same specification will have the same APIs, two scenarios arise one-to-one one-to-many	1- one-to-many classification scenario algorithm of unknown device specification and APIs is unsolved. 2- Assuming the device is already discoverable and connected to the platform or system to have access to the devices' data and gain value out of it
True Self-Configuration for the loT, 2012 (IEEE)	Correlation-Comparison issue	Using Semantic web technologies and SPITFIRE 6lowPan & CoAP	Collecting the raw data of the newly installed sensors compare it with the already deployed one the metadata with highly correlated output is suggested as a template for the new one	This approach cannot integrate undefined devices, is meant only for defined devices that have metadata, TEDS, API.
Introducing MoC Drivers for the Integration of Sensor-Actuator Behaviors in Model-Based Design Flows of Embedded Systems 2016	Deployment gap issue -> faulty error. Deployment gap is a conflict result from mapping dataflow onto event-driven and lack of incorporating standard mechanism	Using the idea of *Model of Computation* (MoC) Introducing framework called MoC-driver	Wrapping the actual sensor/actuator behaviours in an abstract behaviour that provided by the MoC driver then encapsulate it with abstract behavior that generated from standard template for	The approach focused only on Dataflow Process Networks (DPN) of MoC driver while many of implementation of MoC driver left for future work, like Dataflow, Synchronous MoC and event-driven MoC.
IoT architecture for adaptation to transient devices 2020 (Elsevier)	Device and Services conflict	Matching algorithms	When a new IoT device is discovered, the ontology manager examines the non-functional properties and compares them with those of existing devices. If no identical device exists, a new instance is created. If an identical device already exists, the existing individual is reused.	Improving functional properties matching time. Most adaptations are executed in less than 0.2 s, with some cases ranging from 0.4 to 0.9 s.
Revisiting Unknown RFiD tag identification in Large-Scale internet of things 2016 (IEEE). ***Bloom-filter** Based Unknown Tag Identification (BUTI)*	Resolving the ***Tag-tag collision*** problem Efficiently. That is how to separate unknown tag from known tag efficiently after collecting the IDs of all tags.	Using scheme called Bloom-filter and mechanism called Indicator vector to let each tag transmit its ID only once to avoid tag-tag collision.	The method deactivate the known tags and collect unknown tags, then collect IDs of unknown tags using Indicator vector with avoidance of tag-tag collision .	Tag-tag collision problem is not totally avoidable due to optimizing the hash function to minimum false positive ratio and also the limited computation and communication capabilities of RFID tags plays a role.

Table.3 Demonstrate the techniques used for adapting Unknown devices

REFERENCES

[1] A. T. Y. D. J. P. Herencia-Zapana, "Reference modelling in support of M&S—foundations and applications," *Springer*, vol. 7, no. 2, pp. 69–82, 2013.

[2] A. Mavrogiorgou and A. Kiourtis, "Complex, Intelligent, and Software Intensive Systems," vol. 611, 2018.

[3] R. Trimananda, J. Chuang, B. Demsky, G. H. Xu, and S. Lu, "Understanding and Automatically Detecting Conflicting Interactions between Smart Home IoT Applications," *ACM Jt. Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, pp. 1215–1227, 2020.

[4] D. N. Mekuria, P. Sernani, N. Falcionelli, and A. F. Dragoni, "Smart home reasoning systems: a systematic literature review," *J. Ambient Intell. Humaniz. Comput.*, no. 0123456789, 2019.

[5] H. Lin and N. W. Bergmann, "IoT privacy and security challenges for smart home environments," *Inf.*, vol. 7, no. 3, 2016.

[6] A. Al-fuqaha, S. Member, M. Guizani, M. Mohammadi, and S. Member, "Internet of Things : A Survey on Enabling," vol. 17, no. 4, pp. 2347–2376, 2015.

[7] G. Atzori, Luigi and Iera, Antonio and Morabito, "The internet of things: A survey," *Comput. networks*, vol. 54, pp. 2787--2805, 2010.

[8] S. Khan, Rafiullah and Khan, Sarmad Ullah and Zaheer, Rifaqat and Khan, "Future internet: the internet of things architecture, possible applications and key challenges," *IEEE*, no. Frontiers of Information Technology (FIT), 2012 10th International Conference on, pp. 257--260, 2012.

[9] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[10] P. López, D. Fernández, A. J. Jara, and A. F. Skarmeta, "Survey of internet of things technologies for clinical environments," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, 2013, pp. 1349–1354.

[11] C. Jaggernauth and D. Gubbe, "An extended SNMP based IoT context-aware model for dynamic adaptability of embedded systems software," *Proc. 2017 20th Conf. Innov. Clouds, Internet Networks, ICIN 2017*, pp. 211–213, 2017.

[12] D. Lohani and D. Acharya, "SmartVent: A Context Aware IoT System to Measure Indoor Air Quality and Ventilation Rate," *2016 17th IEEE Int. Conf. Mob. Data Manag.*, pp. 64–69, 2016.

[13] W. K. Hsieh, W. H. Hsieh, J. L. Chen, and C. Y. Lin, "Self-configuration and smart binding control on IOT applications," *Int. Conf. Adv. Commun. Technol. ICACT*, vol. 2016-March, pp. 80–85, 2016.

[14] V. Miori, D. Russo, and C. Concordia, "Meeting People's Needs in a Fully Interoperable Domotic Environment," pp. 6802–6824, 2012.

[15] M. Collina, G. E. Corazza, and A. Vanelli-Coralli, "Introducing the QEST broker: Scaling the IoT by bridging MQTT and REST," *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC*, pp. 36–41, 2012.

[16] S. K. Mohalik, N. C. Narendra, R. Badrinath, M. B. Jayaraman, and C. Padala, "Dynamic semantic interoperability of control in IoT-based systems: Need for adaptive middleware," *2016 IEEE 3rd World Forum Internet Things, WF-IoT 2016*, pp. 199–203, 2017.

[17] I. Chatzigiannakis *et al.*, "True Self-Configuration for the loT," 2012.

[18] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM Sigmod Rec.*, vol. 33, pp. 65--70, 2004.

[19] S. M. Kim, H. S. Choi, and W. S. Rhee, "IoT home gateway for auto-configuration and management of MQTT devices," *2015 IEEE Conf. Wirel. Sensors, ICWiSE 2015*, pp. 12–17, 2016.

[20] Y. A. Wang, B. Zhu, and G. Y. Li, "Gateway-based semantic collaboration method in SWoT," *Proc. - 2016 Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov. CyberC 2016*, pp. 136–141, 2017.

[21] A. M. Nagib and H. S. Hamza, "SIGHTED : A Framework for Semantic Integration of Heterogeneous Sensor Data on the Internet of Things," *Procedia - Procedia Comput. Sci.*, vol. 83, no. Ant, pp. 529–536, 2016.

[22] M. Elhamshary, Moustafa and Youssef, "SemSense: Automatic construction of semantic indoor floorplans," *IEEE*, pp. 1--11, 2015.

[23] B. Bordel, D. S. De Rivera, and R. Alcarria, "Plug-and-play transducers in cyber-physical systems for device-driven applications," *Proc. - 2016 10th Int. Conf. Innov. Mob. Internet Serv. Ubiquitous Comput. IMIS 2016*, pp. 316–321, 2016.

[24] G. Fortino, A. Guerrieri, W. Russo, V. P. Bucci, and R. Cs, "Agent-oriented Smart Objects Development," pp. 907–912, 2012.

[25] G. Cabri, F. De Mola, and L. Leonardi, "Agent-based plug-and-play integration of role-enabled medical devices," *Proc. - 2007 Jt. Work. High Confid. Med. Devices, Software, Syst. Med. Device Plug-and-Play Interoperability, HCMDSS/MDPnP 2007*, pp. 111–121, 2007.

[26] N. M. do Nascimento and C. J. P. de Lucena, "FIoT: An agent-based framework for self-adaptive and self-organizing applications based on the Internet of Things," *Inf. Sci. (Ny).*, vol. 378, pp. 161–176, 2017.

[27] J. and others Botia, "Collaborative agents framework for the internet of things," *IOS Press*, vol. 13, p. 191, 2012.

[28] W. Fortino, Giancarlo and Guerrieri, Antonio and Lacopo, Michelangelo and Lucia, Matteo and Russo, "An agent-based middleware for cooperating smart objects," *Springer*, pp. 387--398, 2013.

[29] G. Aloi *et al.*, "A mobile multi-technology gateway to enable IoT interoperability," *Proc. - 2016 IEEE 1st Int. Conf. Internet-of-Things Des. Implementation, IoTDI 2016*, pp. 259–264, 2016.

[30] D. S. G. S. H, D. S. L. W. R. M, P. W. K. Ishara, and M. P. H. S. Kumara, "SmartBee," pp. 251–256, 2008.

[31] T. Falck, H. Baldus, J. Espina, and K. Klabunde, "Plug 'n play simplicity for wireless medical body sensors," *Mob. Networks Appl.*, vol. 12, no. 2–3, pp. 143–153, 2007.

[32] N. Matthys, V. Nv, K. L. Man, F. Yang, and D. Hughes, "The PnP Web Tag A plug-and-play programming model for connecting IoT devices to the web of things," pp. 452–455, 2016.

[33] F. J. Maldonado, "Smart Transducer Integrator (STI) for Standardized System-of-Systems Monitoring," *2018 IEEE Int. Instrum. Meas. Technol. Conf.*, no. i, pp. 1–6, 2018.

[34] "http://www.upnp.org/."

[35] G. Chen, J. Huang, and R. Lin, "Network Interface UPnP Mechanism of IEEE 1451 Smart Sensor," 2017.

[36] V. Rajaraman, P. Misra, K. Dhotrad, and J. Warrior, "Poster Abstract : Enabling Plug-n-Play for the Internet of Things with Self Describing Devices," pp. 374–375.

[37] J. Sung, T. Kim, and D. Kim, "Integration of IEEE1451 sensor networks and UPnP," *2010 7th IEEE Consum. Commun. Netw. Conf. CCNC 2010*, pp. 2–3, 2010.

[38] Á. R. Nieves, N. M. Madrid, R. Seepold, J. M. Larrauri, and B. A. Larrinaga, "A UPnP Service to Control and Manage IEEE 1451 Transducers in Control Networks," vol. 61, no. 3, pp. 791–800, 2012.

[39] J. Yao, R. Schmitz, and S. Warren, "A wearable point-of-care system for home use that incorporates plug-and-play and wireless standards," *IEEE Trans. Inf. Technol. Biomed.*, vol. 9, no. 3, pp. 363–371, 2005.

[40] M. Galarraga *et al.*, "Proposal of an ISO/IEEE11073 platform for healthcare telemonitoring: Plug-and-play solution with new use cases," *Annu. Int. Conf. IEEE Eng. Med. Biol. - Proc.*, pp. 6709–6712, 2007.

[41] P. Thongpithoonrat, P. K. McKneely, S. Gumudavelli, D. Gurkan, and F. M. Chapman, "Networking and plug-and-play of bedside medical instruments.," *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, vol. 2008, no. August, pp. 1514–1517, 2008.

[42] O. Rafique and K. Schneider, "Introducing MoC Drivers for the Integration of Sensor-Actuator Behaviors in Model-Based Design Flows of Embedded Systems," pp. 50–59, 2016.

[43] O. Rafique and K. Schneider, "Towards the Standardization of Plug-and-Play Devices for Model-Based Designs of Embedded Systems," pp. 1–4.

[44] I. a N. F. a Kyildiz, J. O. M. I. J. Ornet, G. E. I. Nstitute, and O. F. T. Echnology, "Revisiting Unknown RFiD tag identification in Large-scaLe internet of things," no. December, pp. 58–63, 2010.

[45] C. El Kaed, Y. Denneulin, and F.-G. Ottogalli, "Dynamic service adaptation for plug and play device interoperability," *Proc. 7th Int. Conf. Netw. Serv. Manag.*, pp. 46–55, 2011.

[46] A. Bottaro, Andr{\'e} and G{\'e}rodolle, "Home soa-: facing protocol heterogeneity in pervasive applications," *AcM*, pp. 73--80, 2008.

[47] 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.

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

[49] K. Kesehatan, "No TitleΕΛΕΝΗ," *Αγαη*, vol. 8, no. 5, p. 55, 2019.

[50] "No Title," *https://aws.amazon.com/iot/*. .

[51] "No Title," *https://www. ibm.com/internet-of-things*. .

[52] "No Title," *https://www. smartthings.com/*. .

[53] 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.

[54] M. Shehata, A. Eberlein, and A. Fapojuwo, "Using semi-formal methods for detecting interactions among smart homes policies," *Sci. Comput. Program.*, vol. 67, no. 2–3, pp. 125–161, 2007.

[55] E. Trundle, "Sift," *Prairie Schoon.*, vol. 86, no. 1, pp. 85–100, 2012.

[56] 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.

[57] 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.

[58] S. J. H. Yang, A. S. Lee, W. C. Chu, and H. Yang, "Rule base verification using Petri nets," *Proc. - Int. Comput. Softw. Appl. Conf.*, pp. 476–481, 1998.

[59] 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.

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