Received: 31 March 2020 | Revised: 10 August 2020 | Accepted: 7 December 2020

DOI: 10.1002/cpe.6189

SPECIAL ISSUE PAPER

WILEY

Exploiting If This Then That and Usage Control obligations for Smart Home security and management

Giacomo Giorgi[1] | Antonio La Marra[2] | Fabio Martinelli[1] | Paolo Mori[1] | Athanasios Rizos[3] | Andrea Saracino[1]

[1]Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Pisa, Italy

[2]Security Forge srl, Pisa, Italy

[3]Department of Computer Science, University of Pisa, Pisa, Italy

Correspondence
Andrea Saracino, Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Via G. Moruzzi n.1, 56124, Pisa, Italy.
Email: andrea.saracino@iit.cnr.it

Funding information
H2020, Grant/Award Numbers: 830892, 952652

Summary

In this article we present an application of the Usage Control paradigm to a Smart Home infrastructure, based on a model extension and structured use of obligations. In the proposed extended model obligations are exploited to enforce two different access revocation time, namely revoke and suspend. This increases the policy expressiveness and enable to optimize the resource usage. Furthermore, obligations are exploited to send commands via IFTTT to different interconnected Smart Home devices, to impose safety-relevant behaviors, or to act on policy attributes to implement a self-healing paradigm for revoked sessions. The article is motivated by a parental control use case where deep learning is used in combination with Usage Control to regulate dynamically viewing rights of a smart-TV and interactions with interconnected devices. Accuracy and performance experiments show the effectiveness and feasibility of the proposed work.

KEYWORDS

access control, deep learning, IFTTT, parental control, Usage Control

1 | INTRODUCTION

In the last years, smart devices are increasingly becoming pervasive in all aspects of our daily life. For instance, traditional devices and appliances in our houses are being replaced by their smart versions (e.g., smart-TV, fridge, and washing machine), which provide additional functionalities and improved effectiveness because they are connected to the network. This enables the *Smart Home* paradigm, where distinct and different devices in the same house communicate among them and with other entities through the local network and the Internet, thus being able to support novel services. As a matter of fact, the standalone capabilities of these devices are exponentially increased when they are able to communicate. Each device has, in fact, different sensors and actuators, and has access to different kind of information which can be used collaboratively to provide more complex services. Hence, a number of novel and advanced applications can be envisaged for the Smart Home scenario, spanning from efficient energy consumption management, to comfort enhancement, elderly and disabled people assistance, home safety and surveillance, and parental control.

Besides introducing the aforementioned advantages in terms of availability of a new generation of innovative and advanced services, the automatization of the devices deployed in a house also poses some new security and privacy concerns and challenges. For instance: is a child authorized to switch on the smart-TV or the smart oven? If the smart-TV has been switched on by an adult and it is playing content for adults only, what we would like the Smart Home management system to do if a child comes in front of the smart-TV?

To address the previously mentioned issues, the control system[1] has been already exploited in the Smart Home environment as a control tool able to handle seamless device access management and data protection,[2,3] as well as to detect policy violations occurring while the access is in progress in order to react by executing mitigating actions. The Usage Control system is an extension of the eXtensible Access Control Markup Language (XACML) standard which defines both a language and a reference architecture to perform traditional attribute-based access control

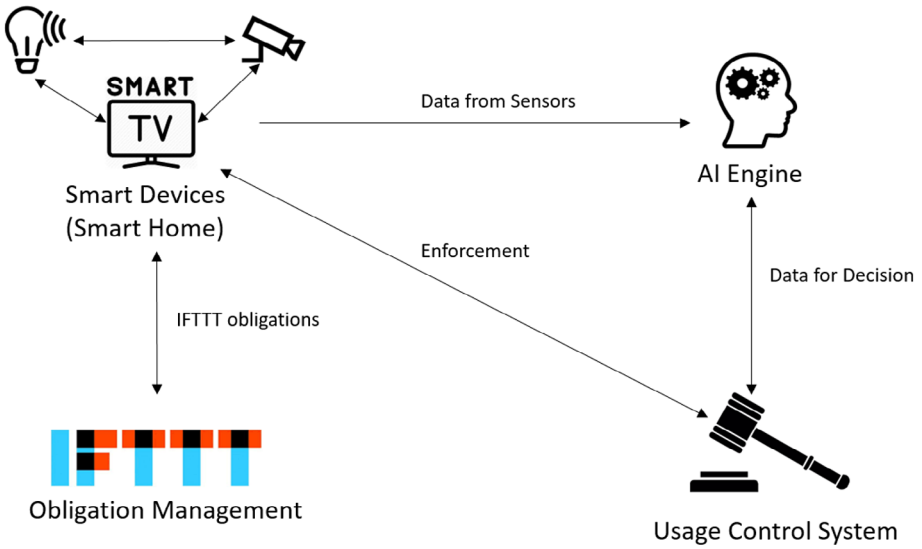


FIGURE 1 Logical architecture of a smart-TV use case

(ABAC). The extension concerns both the XACML language and reference architecture, and it emphasizes the role of obligations, that are mandatory actions that have to be performed in conjunction with the policy enforcement, and the continuity of access decision by evaluating and enforcing policies based on mutable attributes,[4] following the Usage Control (UCON) model.[5] Hence, coming back to the previous example, the Usage Control System, once having given to an adult the permission to switch on the smart-TV and to select content for adults, would be also able to detect when a child comes in front of the TV as a policy violation and, consequently, to ask the execution of an obligation as a countermeasure, such as the smart-TV switch off.

However, although both the XACML language and its extended version, U-XACML, include a basic construct to declare the obligation section within a policy, they do not provide a standard for the description of obligation semantic. While the intention is the one of nonimposing constraints on the format of the represented information, the direct consequence is that obligation management engines have to be developed ad hoc. Instead, having a standard semantic for obligation representation which is meaningful at least in an application macro-environment, would push developers and policy editors to use common expressions to represent and enforce obligations. In particular, in the Smart Home environment, which is a subcase of the Internet of Everything (IoE) environment, it is possible to express obligations as commands for an inter-operable platform used by a multitude of Internet of Things (IoT)/IoE devices, called If This Then That (IFTTT)[1]. IFTTT is a free web-based platform that triggers the execution of actions on IoT devices as a consequence of changes that happen in the specified web-services. Exploiting IFTTT triggers to express obligations will allow the devices receiving such obligations from the Usage Control System as a consequence of the policy evaluation decision to easily execute them, without hard-coding the actual obligation interpretation in the device, or demand the obligation interpretation to a specific applet.

Before our proposal, some effort to extend IFTTT for managing security features have been already performed,[6,7] still not reaching the full functionality of an access control system.

In this article, we present an architecture that exploits the Usage Control System and IFTTT to manage resources and operations in a Smart Home environment. This framework aims at managing security and safety in IoT environment through a single standard interface and policy language. As an example, in Figure 1 we depict a high level view of the proposed framework, applied in a Smart Home environment for a parental control application that will be detailed in the following. As shown, the Usage Control System interacts directly with the smart devices, acquiring data from sensors, possibly elaborated through mechanisms such as artificial intelligence (e.g., for user age estimation), while the IFTTT services are used for obligation management, to control specific actions of the connected smart devices.

We introduce extensions to the standard UCON model to make the paradigm more suitable for a dynamic environment, by handling automated resume of sessions revoked because of policy violations through self-healing policies exploiting IFTTT obligations to force attribute changes needed to satisfy the policy again.

Our proposal aims at enhancing the capabilities of the Usage Control System applied in the Smart Home environment leveraging on the obligation concept, and focusing on security and safety aspects. As a matter of fact, although the Usage Control System already supports obligations, the lack of a standard to express them in Usage Control policies is a relevant obstacle which limits their usage. In the Smart Home environment we overcame this issue and we significantly enhance the capabilities of the Usage Control System thanks to the integration of the IFTTT platform which brings two major advantages: it allows to easily express in the Usage Control policy obligations operating on the IoT devices installed in the Smart Home, and it also simplifies the interactions of the Usage Control System with such devices when asking the execution of actions implementing

[1]https://www.ifttt.com/.

obligations. This would allow the Usage Control System users to exploit the Smart Home devices to execute countermeasures to deal with unauthorized (i.e., undesired and/or dangerous) situations. This integration strengthen the Usage Control System simplifying its adoption in a large number of interesting scenarios. In the following of the article, we will use the previously introduced example concerning a parental control scenario, where a smart-TV interacts with other smart devices in the Smart Home in order to enforce safety policies for smart-TV usage, as reference application to validate the proposed approach.

The contribution of this work are the following:

- We present an architecture based on Usage Control for advanced policy management in a Smart Home environment with interconnected devices.
- We introduce a methodology to express and enforce Access Control and Usage Control obligations by exploiting the IFTTT service.
- We propose an extension to the standard Usage Control model which introduces two new operations to suspend and resume accesses instead of revoking them, which increases the policy expressiveness and allows to optimize the resources usage time.
- We exploit the novel proposed model and the IFTTT-based obligations to implement self-healing policies, where obligations are used after an access revocation (or suspension) to force actions and behaviors meant to bring the system back in a situation compliant with usage policies.
- We present the implementation of a parental control use case, based on the one presented in Reference 8, which exploits the proposed framework to implement a safe and secure usage of a smart-TV and interconnected devices.
- We discuss the design and performance of a novel deep-learning classifier for age estimation, which enables the implementation of the parental control use case by identifying with high accuracy and with a limited performance overhead the number of children and adults currently watching the smart-TV: essential attributes for parental control policies enforcement.
- We evaluate the performance and effectiveness of the proposed approach in both a simulated and in a real use case, demonstrating its feasibility.

This work strongly extends and complete the work presented in Reference 9 and in Reference 8, by proposing a novel application environment and exploiting a novel extension of the UCON model. In particular, the novel contributions w.r.t. such previous papers are (i) the extended Usage Control model which considers the coexistence of the revoke and suspend workflows for a more flexible management of continuous control; (ii) the usage of obligations to implement a self-healing paradigm, to force the execution of actions needed to resume the access in case of policy violations; (iii) the application of IFTTT obligations to a parental control use case with implementation of specific IFTTT applets; (iv) a unique deep learning classifier that simultaneously detects faces and estimates their ages, improving accuracy and performances.

The rest of this article is organized as follows: In Section 2 we report some background information on Usage Control, IFTTT platform, and TV parental guidelines. Section 3 details the proposed architecture and operative workflow. Section 4 details our implementation and discusses the results of the performance analysis. In Section 5 we present a set of related works about security applications of UCON in IoT and XACML obligation expressions. Finally, in Section 6 we offer some conclusions and hint at future directions.

2 | BACKGROUND

In this section, we provide some background knowledge regarding Usage Control, IFTTT platform, and TV parental guidelines.

2.1 | Usage Control

The UCON model[5,10] extends traditional access control models mainly introducing *obligations* and *mutable attributes* in policy evaluation.

Obligations are decision factors that impose the execution of actions as a consequence of the access decision process. For instance, an Usage Control policy could impose that a smart-light must be switched on when a person enters a given room and it must be kept on as long as there are people in such room.

Traditional access control models perform the decision process taking into account *immutable attributes*only, that is, attributes that do not change their values as a consequence of the normal operation of the system, and that is modified only through administrative actions. Mutable attributes, instead, represent features of subjects, resources, and environment that change their values as a consequence of the normal operation of the system.[4] For instance, the number of people in a room is a mutable attribute of the room which changes when a person enters or leaves the room.

Since mutable attributes might change their values during the usage of an object, the UCON model allows the definition of policies (*prepolicies*) which are evaluated before the access (*predecision*), and policies (*ongoing-policies*) which are evaluated continuously during the access to the object (*ongoing-decision*). The *ongoing-decision* phase is meant to detect policy violations that occur while the usage of an object is in progress because the new value of a mutable attribute causes the policy to be not satisfied anymore. For this reason, UCON introduces the concept of access revocation, that is, the capability of stopping ongoing access.

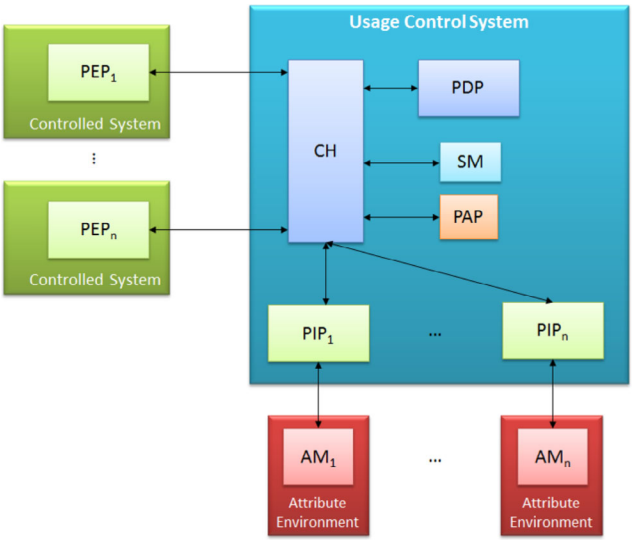


FIGURE 2 Usage control system architecture

This article takes into account the Usage Control System (UCS) presented in a previous work,[1] whose architecture, shown in Figure 2, is based on the XACML reference architecture.[11] As policy language, the UCS exploits an extension of the XACML language, called U-XACML, for dealing with Usage Control features.

In the UCS architecture, the policy enforcement points (PEPs) embedded in the controlled system intercept the execution of security-relevant operations, and invoke the context handler (CH), which is the front-end of the UCS. The policy information points (PIPs) are the components invoked by the CH to retrieve the current values of the attributes required by the policy decision point (PDP) for the execution of the decision process, that is, to evaluate the policy retrieved from the policy administration point (PAP). Each specific scenario where the UCS is exploited has its own set of attribute managers (AMs) which manage the attributes required for the policy evaluation. Consequently, a number of policy information points (PIPs) must be properly configured for each specific scenario in order to be able to interact with such AMs.

The phases of the Usage Control decision process are regulated by the interactions between the PEPs and the UCS as follows (derived from Reference 12):

TryAccess : is the *predecision* phase, which begins when the *TryAccess* message is sent by the PEP to the UCS because a subject requests to execute an access. The *predecision* phase finishes when the UCS sends the response to the PEP. The possible responses are: *PermitAccess*, to allow the access, or *DenyAccess*, to reject the access request;

StartAccess : this interaction triggers the *ongoing-decision* phase, which begins when the *StartAccess* message is sent by the PEP to the UCS because the access has just started, and causes a continuous evaluation of the *ongoing-policy*. In case of policy violation, a *RevokeAccess* message is sent to the PEP, causing the access to be stopped, otherwise the *ongoing-decision* phase is executed until the access is not terminated by an *EndAccess* message;

RevokeAccess : this is part of the *ongoing-decision* phase. Every time an attribute changes its value the policy is evaluated again and, in case of violation, a *RevokeAccess* message is sent by the UCS to the PEP, causing the access to be stopped;

EndAccess : sent by the PEP to the UCS when ongoing access terminates normally.

The workflow of the Usage Control decision process is the following. When the subject *s* tries to execute a security relevant action *a*, the PEP suspends the execution of *a* and retrieves the information related to the access context (subject and resource IDs, etc.). The PEP sends the *TryAccess* message with the data previously collected to the UCS, which performs the *predecision* process and returns the result of the *prepolicy* evaluation to the PEP, which enforces it. If the execution of *a* is permitted, the PEP sends the *StartAccess* message to the UCS as soon as *a* is started, and the UCS executes the first part of the *on-decision* phase, thus performing a first evaluation of the *ongoing-policy*. If the policy is violated, the UCS sends the *RevokeAccess* message to the PEP, in order to take proper countermeasures. Instead, if the policy is satisfied, the execution of *a* continues, and each time an attribute changes its value, the second part of the *on-decision* phase is performed, thus evaluating the *ongoing-policy* again. If the policy is violated, the UCS sends the *RevokeAccess* message to the PEP, in order to take proper countermeasures. Otherwise, no message is sent and the execution of *a* continues.

2.2 | IFTTT

IFTTT derives its name from the programming conditional statement "If This, Then, That." IFTTT provides a software platform that connects apps, devices, and services from different developers in order to trigger one or more automation involving those apps, devices, and services.

TABLE 1 Comparison of IFTTT and its alternatives

Name	Cost	Connections	Apps/devices	Number of services	Access
IFTTT	Free	One-to-One	Yes/yes	>600	App/bowser
Microsoft Flow	Free/paid	Multiple	Yes/no	226	App/browser
Zapier	Free/paid	Multiple	Yes/no	>1500	Browser
Yonomi	Free	Multiple	No/yes	200	App
Stringify	Free	Multiple	Yes/yes	70	App/browser
Workflow	Free	Multiple	Yes (iOS)/no	N/A	App

IFTTT creates chains of simple conditional statements that are called *Applets* and are triggered by changes that happen within web-services.

IFTTT consists of the following structure:

`Services`: The basic building block of IFTTT. They describe data and actions controlled by an application programming interface (API), and each service has a specific set of *Triggers* and *Actions*.

`Triggers`: The *"This"* part of the *Applet*, causing thus the triggering of the *Action*.

`Actions`: The *"That"* part of the *Applet* which is the result of a *Trigger*.

`Applets`: The composition of a *Trigger* and an *Action*.

An advantage of IFTTT is that it can work with various platforms and devices in IoT. A drawback of IFTTT is that it allows only a single *Trigger* and a single *Action* in the same *Applet*. The same *Trigger* can be used for other *Actions* but not inside the same *Applet*.

There are various competitors of IFTTT and in Table 1 we see a summary of the pros and cons of them. More in detail, the biggest competitor of IFTTT is `Microsoft Flow`. It is free for up to 750 runs/month, it can allow multistep connection and works with many apps including Gmail, Facebook, and so on and can be accessed either via apps or browsers. The cons are that, up to now, the apps are around 226 and it does not work with physical devices. But, it also supports Do/While and For/Each loop whereas IFTTT supports only If conditionals. Hence, it is more difficult and complicated to operate. Another example is `Zapier` that works only with apps and not physical devices. It allows multistep connection between devices and has a simple free usage up to 100 runs/month and also paid plans of use. On the contrary, `Yonomi` works only with physical devices and not with apps, but allows multistep connection between the about 100 compatible devices, it is free and focuses more on home applications. Furthermore, `Stringify` is also free and can host multiple connections but has only about 70 services and cannot be accessed via a browser like IFTTT but only via an app. Finally, there is also `Workflow` that works only with iOS apps and allows multiple-step systems. It is again free but there is not a list of compatible services but only the most famous iOS apps are used like Safari, Photo Gallery, Facebook, and so on.

2.3 | TV parental guidelines

Multimedia content, such as movies and TV programs, are classified according to standards in order to advise the intended audience about features that might be considered dangerous or offensive to a set of spectators. In this work we refer to the American standard for parental guidelines [2], where the guidelines are based on two parameters, namely the *Audience* and the *Content Label*. The *Audience* parameter may assume one of the following values:

- (Y) for all children: This means that the program is not expected to frighten younger (2–6-year-old) children.
- (Y7) older children: This program is designed for children at the age of 7 and above. Some of these programs may also have the Fantasy Violence (FV) *Content Label* in case that fantasy violence is more combative than in other programs.
- (G) General Audience: Program for all ages. Even if this program is not specifically designed for children, most parents may let younger children watch this program unattended.
- (PG) Parental Guidance: many parents may find this program unsuitable for younger children (the theme may call for parental guidance).
- (14) Parents Strongly Cautioned: many parents may find this program unsuitable for children under 14. They are also urged against letting children unattended.
- (MA) Mature Audience: Programs specifically designed to be viewed by adults. *Content Label* indicates if a show might contain violence, sexually explicit content, mature language, and so on.

[2]http://www.tvguidelines.org/resources/TV_Parental_guidelines_Brochure.pdf.

Each program may have assigned one or more *Content Labels* as follows:

- (D): Suggestive dialogue
- (L): Coarse or crude language
- (S): Sexual situations
- (V): Violence
- (FV): Fantasy Violence (children's programming only)

The presence of one or more *Content Labels* among D, S, and L, defines whether a program or a content should have intended audience is PG, 14 or MA.

3 | METHODOLOGY

In this section, we report the description of the proposed framework for enforcing UCON policies in the Smart Home scenario implementing obligations via IFTTT. As a matter of fact, as shown in Section 2.1, the UCON model covers obligations, but the extension of the XACML policy language that has been defined for dealing with UCON features, U-XACML,[13] does not cover a language to express such obligations. The main reason is that the obligations that are enforced by the UCON policies are typically very dependent on the application scenario where the policy is enforced, and hence it is difficult to define a standard language for dealing with all kinds of obligations. Instead, in some application scenarios, such as the Smart Home one, specific languages already exist for expressing scenario-specific action execution commands. For this reason, in the Smart Home scenario, we decided to exploit IFTTT as a language to define obligations in UCON policies.

3.1 | Extended Usage Control model

In the original UCON model (described in Section 2.1), when a policy violation is detected in the *ongoing-decision* phase, the corresponding access is revoked. This work considers an extended version of the UCON model,[14] which pairs the standard revoke operation with a softer usage revocation, named *SuspendAccess*. In fact, some scenarios only need to suspend the access as long as the Usage Control policy is violated, and the access should be resumed as soon as the Usage Control policy is satisfied again.

To describe the proposed extension we follow a similar approach to,[12] where a set of *Usage Control actions*are used to describe the actions performed by the subject and the ones performed by the UCON system to implement the Usage Control decision process. Hence, given that the triple (s, o, r) which represents an access request where a subject s wants to access an object o through an operation that requires the right r, we extend the UCON model as follows:

- *TryAccess*(s, o, r): performed when the subject s requests a new access (s, o, r). This triggers the execution of the *predecision* phase.
- *PermitAccess*(s, o, r): performed by the system as a result of the *predecision* phase, when the evaluation of the policy allows the access request (s, o, r).
- *DenyAccess*(s, o, r): performed by the system as a result of the *predecision* phase, when the evaluation of the policy forbids the access request (s, o, r).
- *StartAccess*(s, o, r): performed when the access (s, o, r) has been allowed by the *PermitAccess*(s, o, r) and it has actually started. This triggers the execution of the *ongoing-decision* phase.
- *EndAccess*(s, o, r): performed when the subject s terminates an ongoing access (s, o, r).
- *RevokeAccess*(s, o, r): performed by the system as a result of the *ongoing-decision* phase when the policy is violated and it requires to block the usage of a resource.
- *SuspendAccess*(s, o, r): performed by the system as a result of the *ongoing-decision* phase when the policy is violated and it requires to suspend ongoing access (s, o, r).
- *ResumeAccess*(s, o, r): performed by the system as a result of the *ongoing-decision* phase when the policy requires to resume an access (s, o, r) previously suspended by the *SuspendAccess*(s, o, r) action.

As shown in Figure 3, with respect to the traditional UCON model, two new actions have been added to the Usage Control decision process: *SuspendAccess* and *ResumeAccess*. The *SuspendAccess* action, instead of definitely terminating ongoing access, simply suspends it. This means that

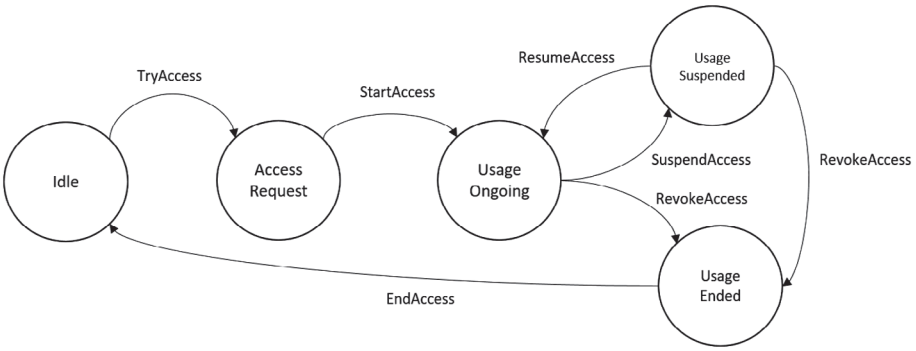


FIGURE 3 State machine of the extended UCON model

the *ongoing-decision* phase is still performed for that access, until either the subject decides to terminate it, or the UCON system decides to resume or definitely terminate such access. The *ResumeAccess* is issued for suspended access when, during the *ongoing-decision* phase, the evaluation of the policy determines that the access is authorized again and, consequently, it must be resumed.

This model helps in maximizing the resource usage time when an *ongoing-policy* violation only requires a temporary suspension of the access until the policy is satisfied again. In fact, when the *ongoing-policy* is violated, the standard UCON model terminates the resource usage through a *RevokeAccess*. Consequently, the access can be continued only if the user explicitly issues a new *TryAccess* request, given that the request is matching the *prepolicy* and the *ongoing-policy*. However, it is possible that mutable attributes already were back in a configuration matching the policy before the actual time in which the new *TryAccess* is performed. In such a case, the time elapsed between the attribute changes and the moment in which the new *TryAccess* is performed by the user is wasted usage time. Formally, naming t_c the time in which the mutable attributes values match again the ongoing policy conditions, and naming $t_{ta} > t_c$ the time in which the *TryAccess* is performed, the time $\Delta_w = t_{ta} - t_c$ is the amount of usage time that goes unused and wasted. The introduction of the *ResumeAccess* reduces Δ_w.

Differently, from what proposed in Reference 14, the extended model we propose in this article still includes the *RevokeAccess* action to increase the expressiveness of the policies. In fact, this model allows to differentiate among (i) violations that require that the whole Usage Control policy is reevaluated (i.e., both the *prepolicy* and the *ongoing-policy*) to continue the access (e.g., active authentication is required each time the resource is accessed), and (ii) violations which require that only the *ongoing-policy* is reevaluated to decide whether to resume the access (e.g., monitoring the temperature in the room, suspending the usage if it goes over the threshold and resuming it as it changes back to the acceptable range). For (i), the access revocation has to be handled with the *RevokeAccess*, which triggers the *EndAccess* that definitely terminates the access. In (ii), a *SuspendAccess* can be used instead, thus mutable attributes will be continuously monitored and the *ongoing-policy* reevaluated. When the reevaluation returns a permit, the *ResumeAccess* will be invoked enabling the usage again.

At the implementation level, the management of the two different strategies to react to *ongoing-policy*violations can be handled through policy obligations. In fact, in U-XACML an access revocation is enforced as the result of a `DENY` decision returned by the PDP as a result of the evaluation of an *ongoing-policy*. Thus, an obligation which specifies if this deny decision should imply an access revocation or an access suspension should be included in the policy. This obligation is thus returned to the PEP together with the DENY decision, driving the PEP behavior. In case the obligation is forcing a *RevokeAccess*, the PEP will terminate the resource usage and will invoke an *EndAccess* as for the usual UCON flow, which in turns blocks the attribute monitoring and policy reevaluation for that access. In case the obligation implies a *SuspendAccess*, the resource access is also terminated or put on hold (depending on the resource type). The *EndAccess* is not invoked, and attributes are still monitored in such a way that, every time they change, the *ongoing-policy* is reevaluated for that access, until a permit decision is returned, or in alternative a definitive revoke due to a different policy rule returning deny.

3.2 | Use cases

This section presents three different applications of our framework to show the flexibility of our model as it adapts to several different context.

3.2.1 | UCON obligation via IFTTT in a smart-greenhouse

The first example consists of advanced management through policy enforcement for remote urban farming in smart-greenhouses.[9] The representation of the scenario is shown in Figure 4. The greenhouse is composed of a set of smart-sensors, which in our specific scenario are humidity and temperature sensors, The UCS by monitoring the greenhouse attributes, can provide access to the smart-heating device to perform scheduled heating of the plants. The obligations are targeting the watering schedule of the smart-watering device which is responsible for watering the plants. The

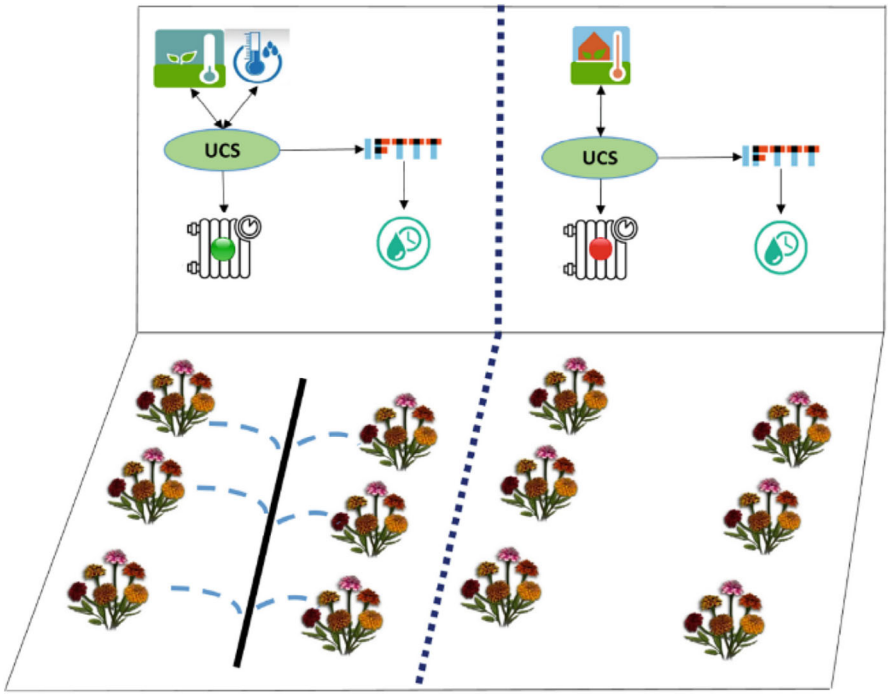


FIGURE 4 UCON obligation via IFTTT in a smart-greenhouse installation

goal in this use case is to maintain the temperature and humidity levels of the greenhouse to the desired values via controlling the smart-heating with UCS and change the schedule of the smart-watering device with IFTTT via the obligations coming from UCS. The PEP component is embedded in the smart-heating device and it asks for permission to operate on the smart-greenhouse. It communicates with the UCS, initially with a StartAccess action, in order to request access to operate. The UCS monitors the values of the humidity, temperature, and it takes a decision on the basis of the request and the policy set. The response to the StartAccess sent by the UCS of a successful request contains an obligation including a schedule for the smart-watering device with the operative settings. The PEP will enforce the obligation by sending the web-request to the IFTTT platform that acts directly on the smart-watering device, setting thus the received schedule. The UCS performs a continuous reevaluation of the policy monitoring the attributes received by the smart-sensors. Thus, when occurring a policy violation, for example, the weather is too hot or the smart-heating device is operating for too long, the UCS performs the RevokeAccess to the PEP to stop operating. Such action includes an obligation containing the new schedule for the smart-heating.

3.2.2 | UCON obligation via IFTTT in a smart-office

Another use case is smart management through policy enforcement for controlling a smart-office.[9] The goal of this scenario is to continuously monitor the presence of people inside an office, the air quality of the room while they are inside and the temperature. The representation of this scenario is shown in Figure 5, where there are shown two cases. On the left, there is an example with a few people inside the office. The PEP is installed in the smart-heating device and request for access from the UCS to operate as scheduled. In the response of the UCS after the StartAccess there is an obligation for the IFTTT to force stop the ventilation system. The other example depicted In Figure 5 represents a situation with too many people inside the office. Thus, due to the not comfortable condition inside the office, there is a policy violation. Then, independently from the schedule of the smart-heating, UCS issues the RevokeAccess action to force the smart-heating device to stop. The obligation that comes together with the RevokeAccess, forces, through the execution of the IFTTT Applet, the smart-ventilation system to start operating. In this example, the first obligation comes with the response after a successful StartAccess and forces through the IFTTT the smart-ventilation system to stop. The second obligation comes with the RevokeAccess and forces through the IFTTT the smart-ventilation system to operate.

3.2.3 | UCON obligation via IFTTT in a Smart Home

The third use case, which is used as a reference scenario to validate the proposed framework, concerns a Smart Home where the usage of a smart-TV is regulated through our framework. As a matter of fact, the smart-TV allows the people in the house to easily access a large number of multimedia contents, being some of them addressed to adults only (the ones that fall in the category MA according to the classification reported in Section 2.3),

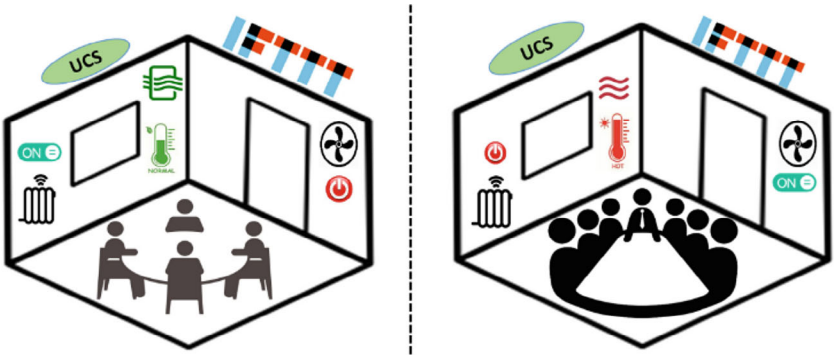


FIGURE 5 UCON obligation via IFTTT in a smart-office installation for monitoring air quality

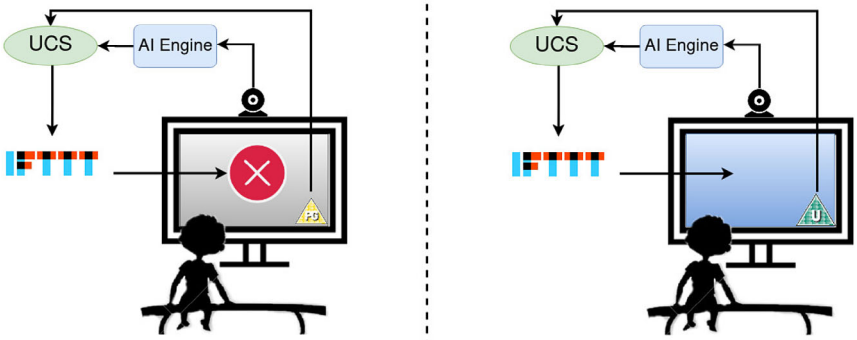


FIGURE 6 UCON obligation via IFTTT in a smart-TV installation for parental control

or requiring the presence of an adult for a child to safely watch them (categories 14 and PG). Hence, an issue that arises with smart-TVs is that children or minors can very easily watch contents (e.g., live programs or on-line videos) that are not meant for them. The relevance of the problem is confirmed by the fact that some smart-TVs and some apps or content providers already include some solutions to enforce parental control. However, these solutions are mainly based on manual settings of channels, programs, or time slots that should be accessible or not to the children, where the forbidden contents are generally protected through a PIN or a password. This makes the parental control service cumbersome to use since adults are not willing to insert a password/PIN any time they wish to watch a program or content not meant for children. Moreover, these approaches have limited flexibility, because they do not consider, for example, those contents which are allowed to minors if watched in the presence of an adult. For this scenario, is also not considered the case in which the adult can temporarily leave the minor to watch the content, without pausing it. Figure 6 shows the smart-parental control use case. The attributes that must be taken into account by the policy are represented by the ages of the people in front of the smart-TV which are detected by the artificial intelligence engine that analyzes the video stream of the camera embedded in the smart-TV and the TV program recommendation provided by the TV content metadata. The PEP is embedded in the smart-TV to enforce the decision provided by the UCS through IFTTT obligation. The UCS, monitoring the aforementioned attributes, takes a decision on the bases of the request and the policy set. If the decision result returns a PERMIT, the UCS response does not contain any IFTTT obligation, whereas if a violation occurs, the UCS response returns a DENY and embed an IFTTT obligation to be executed on the PEP that performs a revocation or a suspension of the TV content.

3.3 | Relevant Usage Control policies

Considering the aforementioned Smart Home scenario, we define the following set of Usage Control policies to regulate the usage of the smart-TV:

Policy 1: The user can watch the TV IF the *program classification* is one-of {Y, Y7, G} OR the *program classification* is PG and the *number of adults* is greater than 0 OR *program classification* is one of {14, MA} and the *number of children* is equal to 0. On PERMIT switch on the light if ambience light is not sufficient.

This is a *prepolicy* to ensure that the right of watching the TV is granted according to the parental guidelines on the streaming program. Furthermore, the policy contains an obligation used to enforce the right lighting conditions in the room to not fatigue the eyesight of spectators, by interacting with the smart-lights in the room, if needed.

Policy 2: As long as the program classification is one-of {Y, Y7, G}, the vision can continue.

Policy 3: As long as the program classification is PG and the *number of adults* is greater than 0, the vision can continue. In case the program classification is PG and the *number of adults* is equal to 0, the vision is *suspended* and the adult has to choose one of: (i) change program, (ii) wait for adult presence, (iii) switch off the TV.

Policy 4: AS LONG the program classification is one-of {14, MA}, and the *number of children* is equal to 0, the vision can continue. In case the program classification is one-of {14, MA}, and the *number of children* is greater than 0, the vision is *suspended* and the adult has to choose one of: (i) change program, (ii) wait for children absence, (iii) switch off the TV.

Policy 2 is an *ongoing-policy* which simply allows to continue the vision of the program as long as its classification is Y, Y7, or G.

Policy 3 is an *ongoing-policy* which allows to continue the vision of a PG classified program only if at least an adult is watching such program. This policy suspends the right visualize the program when no adults are in front of the TV to avoid that a child is left alone watching a PG classified program. In such a case, through the obligation enforced via IFTTT, a notification is sent to the TV owner smartphone, who can either change the program, switch off the TV, or do not act. Changing the program will imply the mutation of the *program classification* attribute value, hence a policy reevaluation. For instance, if the new program is classified Y, Y7 or G and no other attributes have changed, the reevaluation will allow the streaming of the new program, hence the usage of the smart-TV will be granted again.

Policy 4 is an *ongoing-policy* which suspends the right to watch the TV if a child pops up while the adult is watching a program classified 14 or MA. Similarly to the previous case, through the obligation enforced via IFTTT, a notification is sent to the TV owner smartphone, who either change the program, switch off the TV, or do not act.

Policy 5: IF the *current time* is different from allowed viewing hours and the *number of children* is at least 1, the vision of the TV is *revoked* or *denied*.

This policy contains a preauthorization and ongoing condition to define ranges of time in which the child/children is allowed to watch the TV, either forbidding the TV usage out of allowed time ranges or stopping the usage once the allowed watching time range passes.

3.4 | Architecture

The aim of this article is to evaluate the adoption of the UCON framework to implement a parental control system applied on a smart-TV. The main idea is to react to policy violations through UCON obligations enforced via IFTTT. To this aim, we propose the architecture in Figure 7, which is composed by four components described in the following.

- **Smart-TV**: the main component of the architecture is a smart-TV offering internet connectivity and a range of applications (apps) for the TV program managing. In our scenario, the smart-TV embeds a camera used to acquire the scene in front of the TV. Such camera is used by the *video*

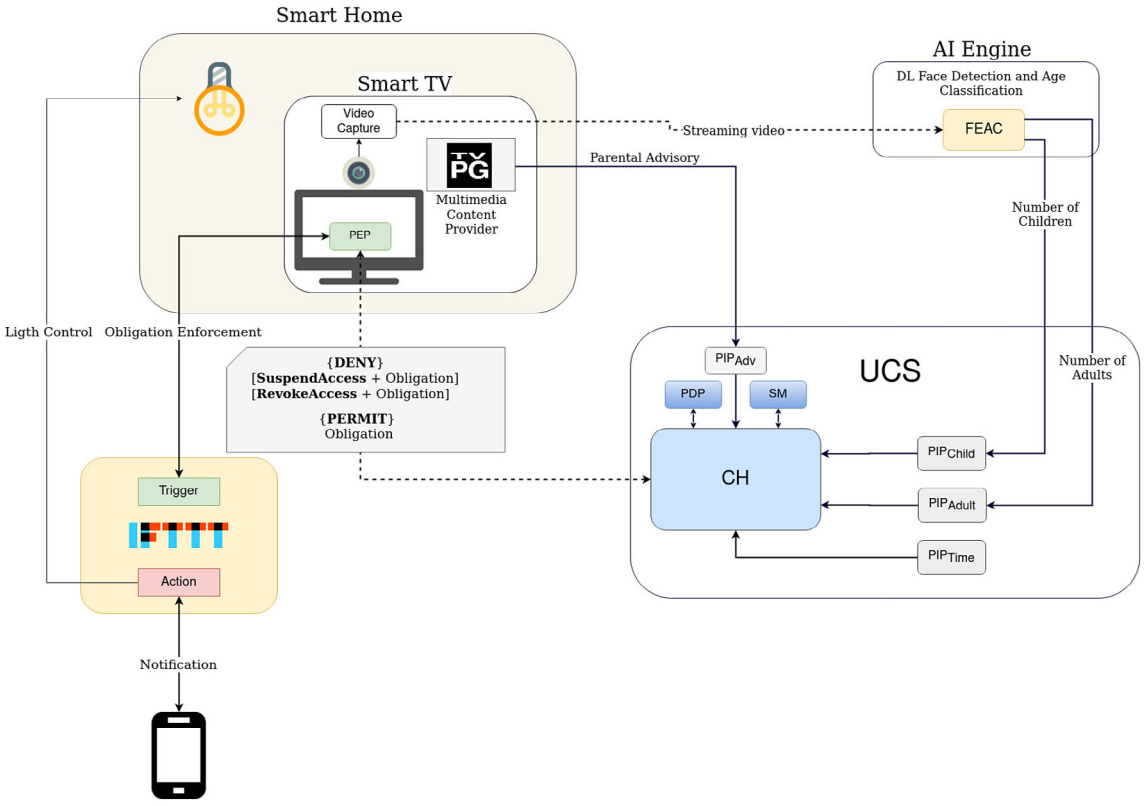


FIGURE 7 Architecture of the smart parental control use case

capture component to transmit the live video to the AI Engine. The built-in app that manages the smart-TV retrieves the TV program recommendation (*Audience* and *Content Label*) directly from the content metadata when available (natives in Android TVs), or querying external databases with content information such as the Internet Movie Database (IMDb). Such metadata are continuously sent to the PIP_{Adv} located in the UCON framework. Finally, a *PEP* is embedded in the smart-TV to interact with the UCS, that is, to send access requests to the UCS and to receive the related responses. If the response of the UCS includes an obligation, the PEP enforces it directly on the smart-TV or by activating the Trigger of the corresponding IFTTT Applet.

- **AI Engine**: the aim of the AI Engine server is to analyze the received video stream related to the environment in front of the smart-TV. The analysis is performed to detect the faces of the people in front of the TV and estimate their ages. Such activity exploits a real-time Deep Learning object detector trained to simultaneously detect faces and their ages on a video stream. The output of the network is modified to inform in a continuous way the PIP_{child} and PIP_{adult} components about the number of children and adults detected in front of the smart-TV camera.
- **UCS**: the Usage Control System is invoked by the PEP embedded in the smart-TV to trigger the *predecision* phase every time the smart-TV is switched on. To evaluate the *prepolicy*, PIP_{Adv} extracts the TV program recommendation (*Audience* and *Content Label*) from the message received from the smart-TV, while PIP_{Child} and PIP_{Adult} extract the number of children and adults, respectively, from the AI Engine. These PIPs also notify the UCS when the value of these attributes change, in order to trigger the *ongoing-decision*phase. Exploiting these attributes, the PDP evaluates the *prepolicy* and the *ongoing-policy*according to the Usage Control decision workflow, and the result is sent to the PEP located to the smart-TV. This result could embed obligations to be executed by the PEP, which specify actions to be triggered through the IFTTT, for example, switch on the TV back light. However, when the evaluation of the *ongoing-policy* returns DENY, obligations are also used to communicate to the PEP whether the access must be revoked or simply suspended.
- **IFTTT component** The IFTTT component is the service that manages the execution of actions controlled through web based requests. It includes a *Trigger* component, which is invoked by the PEP for the enforcement of obligations. Each web request done by the PEP calls such trigger that invoke a specific *Action*, that in our scenario can be a message notification on the TV owner smartphone or a lighting control in the room.

3.5 | Workflow

The complete workflow of our framework is presented in Figure 8. The figure describes the communication between the PEP and UCS implementing the Usage Control process workflow, exploiting the Usage Control actions defined in Section 3.1.

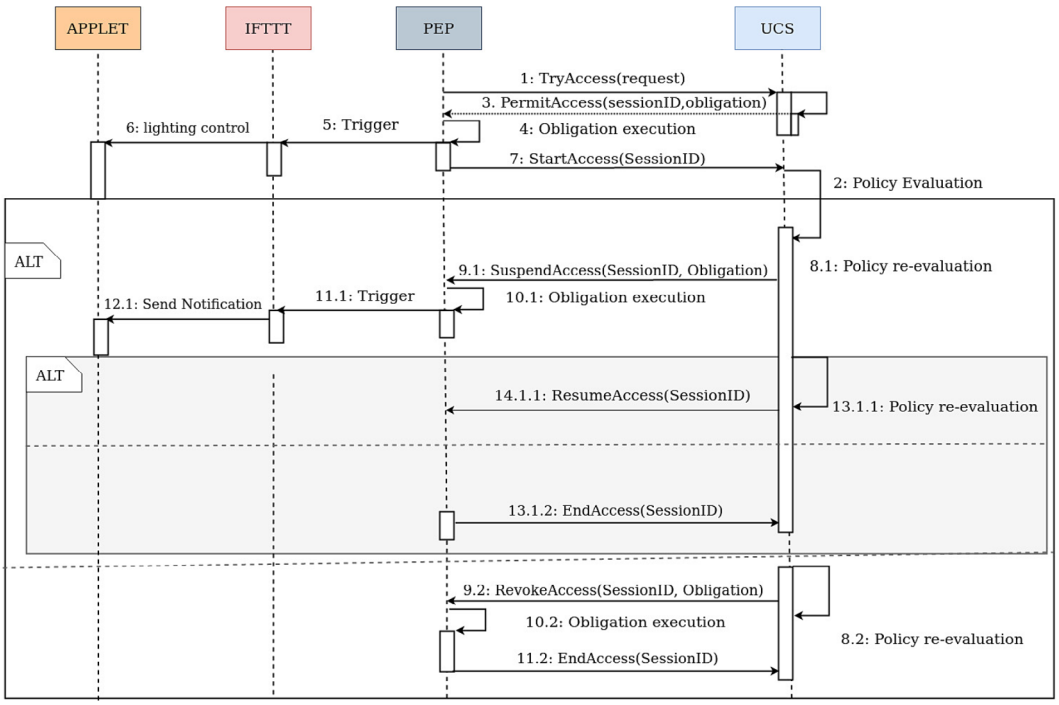


FIGURE 8 Policy enforcement workflow

As shown in Figure 8, the PEP primarily initiates communication with the UCS by performing the *TryAccess* action for the evaluation of the request (1:TryAccess(request)). The CH component of the UCS receives the request and, consequently, collects the current values of the attributes through the PIPs (PIP_{child}, PIP_{adults}, $PIP_{advisory}$, PIP_{time}). The request and the current attribute values are sent to the PDP which evaluates the *pre-policy* (2:Policy Evaluation). Let us suppose that the result of the policy evaluation is PERMIT. In this case, a unique session ID is assigned to this access and the PEP is informed that the access is granted (3:PermitAccess(sessionID,obligation)). The PEP executes the obligations included in the UCS response (4:Obligation execution) by triggering the IFTTT action (5:Trigger). The IFTTT server forwards the command to the applet which interacts with the related smart object. For instance, following Policy 1, the IFTTT server must enforce the right lighting condition by interacting with the smart-lights in the room, if needed (6:lighting control). After the obligation enforcement, the PEP starts the actual usage of the resource, and notifies the UCS by performing the *StartAccess* action for the session with the specific ID (7:StartAccess(SessionID)). Consequently, the UCS starts the continuous evaluation of the *ongoing-policy*(8.1:Policy reevaluation/8.2:Policy reevaluation). While the usage session is in progress, the PIPs monitor the value of attributes and notify the UCS when the value of an attribute changes, thus triggering the PDP for a reevaluation of the *ongoing-policy*. As long as the policy is satisfied, the UCS does not interact with the PEP. In case of policy violation, the policy itself specifies to the UCS to perform either the *SuspendAccess* suspending the ongoing access to the resource (9.1:SuspendAccess(SessionID, Obligation) or the *RevokeAccess*, definitely terminating such access (9.2:RevokeAccess(SessionID, Obligation)). Let us suppose to enforce Policy 3. In case of policy violation, the access is suspended and the obligation is to notify the TV owner. Hence, the PEP enforces such obligation (10.1:Obligation execution) by communicating with the IFTTT server (11.1:Trigger) which, in turn, invokes the applet to send the notification (12.1:Send Notification). Since the access is suspended, the mutable attributes will be continuously monitored, in order to detect attribute changes. If the user changes the TV program or video, the value of the attribute returned by PIP_{Adv} changes, the policy is reevaluated (13.1.1:Policy reevaluation) and, if the decision is PERMIT, the UCS sends to the PEP the *ResumeAccess* action (14:1.1:ResumeAccess(SessionID)), enabling the usage of the resource again. Instead, if the smart-TV is switched off, the EndAccess message is sent by the PEP to the UCS (13.1.2:EndAccess(SessionID)), which terminates the session.

3.6 | Implementation details

In this section, the implementation details related to each component of the framework are provided. The implementation of the AI Engine component is described specifying details related to the training approach adopted. In addition, details about the implementation of the testing scenario and the policy enforcement on smart-TV are given.

3.6.1 | Face detection and age estimation

Unlike the implementation presented in Reference 8, where the face detection and the age estimation were separated tasks executed in a pipeline composed by a face extraction layer (FEL) and an age estimation layer (ACL), the new implementation is composed by a single network which simultaneously localizes the faces as well as estimates the ages (face extraction and age classification, FEAC). The component exploited for the detection and estimation is Yolo V3,[15] the state of the art deep learning network in real-time object detection. By analyzing an image or a frame, it is able to detect different objects depending on the learning activity previously done. It takes images as input, passes it through the neural network, and get a vector of bounding boxes and class predictions as output. Yolo V3 uses DarkNet-53 to make features detection followed by convolutional layers. Darknet-53 is a 53 layers Convolutional Neural network trained on ImageNet dataset[16] mainly compose of 3×3 and 1×1 filters with skip connections like the residual network in ResNet.[17] To simultaneously obtain face localization and age estimation, we trained our classifier exploiting the concept of transfer learning, following the same methodology explained in Reference 8. As shown in Figure 9, we started with a darknet-53 pretrained model that exploits the weights learned on the imageNet dataset which contains 1000 classes. Starting from the fact that the first layers of the network (convolutional layers) learn lower-level image features and the last fully connected layers evaluate the features to determine to which class they belong, the main idea is to start with some knowledge about object detection (imageNet dataset), then retrain the last fully connected layer using the specific dataset suitable for estimating ages and detect faces. As a transfer learning dataset, we exploited IMDb-WIKI dataset.[18] It is the largest public dataset of face images with age and gender labels. It is gathered using the most popular 100,000 celebrity images from IMDb and Wikipedia websites. In total the dataset is composed of 460,723 face images from 20,284 celebrities from IMD and 62,328 from Wikipedia, thus 523,051 in total. As shown in Figure 10 and explained in Reference 8, the dataset is unbalanced with missing data in the range [0–15] and [70–100] ages. However, the dataset contains a large amount of data useful to learn lower level features to detect faces and estimate ages in an image frame. For that reason, starting from the Yolo V3 pretrained on ImageNet, we transfer its learning on the IMDb-WIKI dataset to adapt the network on the new tasks (face detection and age estimation of 101 classes). Thus, we fine-tuned the obtained model on a smallest but balanced dataset in order to get higher accuracy on the minority classes. As balanced dataset we exploited the Audience Db[19] dataset, which contains 26,580 face images from 2284 subjects divided in 8 different age ranges, equally distributed as shown in Figure 11. The dataset has been filtered, removing images with

FIGURE 9 Training methodology

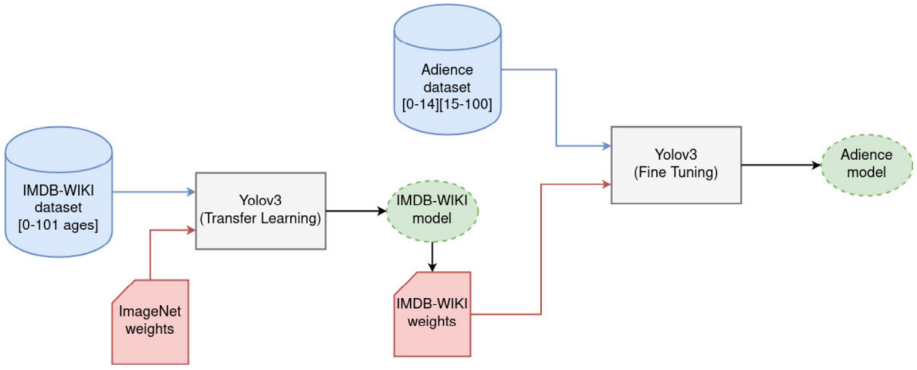


FIGURE 10 IMDb-WIKI age distribution

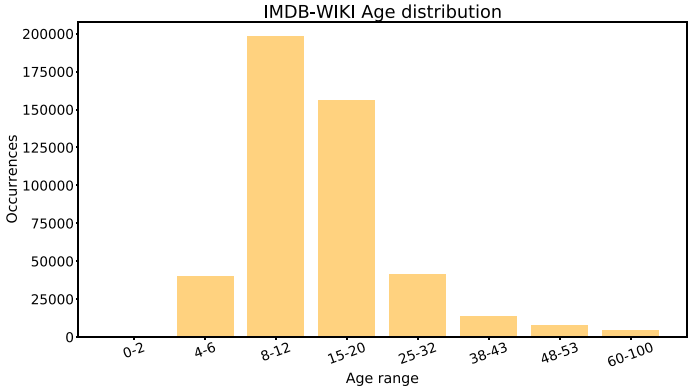
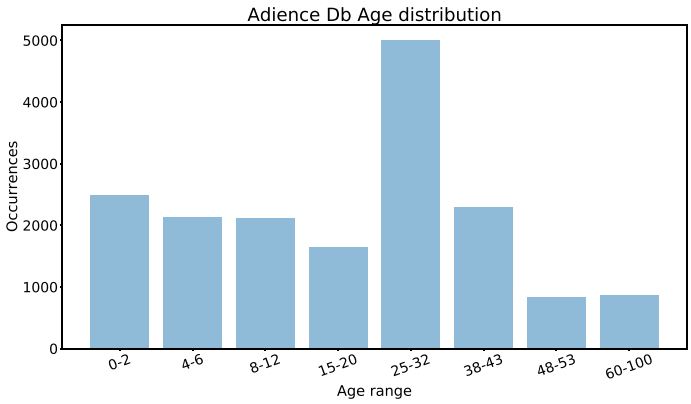


FIGURE 11 Audience Db age distribution



multiple faces belonging to different age ranges or images wrongly labeled, obtaining thus a dataset consisting of 5.949 faces in the age range [0–14] and 5.937 faces in the age range [15–100].

3.6.2 | Testbed implementation and smart-TV simulation

The smart-TV can be considered by definition a component of a Smart Home environment. In fact, together with offering a large set of advanced entertainment features, being connected to the home network, it can easily communicate with other devices connected that are part of the Smart Home. Hence, it is possible to exploit the different functionalities and computational power of the device for providing additional joint functionalities and improving the whole system performance.

A preliminary version of this architecture has been presented in Reference 8, our previous work more focused on the deep learning aspects. The architecture has been refined, by upgrading the functionalities of the components and by introducing the IFTTT-based obligations in the workflow. More in details, as anticipated, the FEL and ACL have been completely reimplemented with a new, single network logic, the training strategy has been completely revised and this brought better performance in terms of accuracy and overhead. In addition, we have considered the interaction

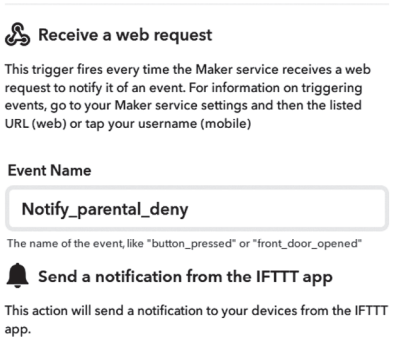


FIGURE 12 IFTTT trigger part

with more external devices (e.g., the Smart-Light to ensure lighting conditions which do not harm the viewer sight). Furthermore, the presence of IFTTT-based obligation, strongly improves the functionalities of the smart-TV by (i) considered the interaction with more external devices (e.g., the Smart-Light to ensure lighting conditions which do not harm the viewer sight), and (ii) by enabling self-healing policy for a smarter reaction and better control of the parental functionalities.

In particular, since from the point of view of performance the smart-TV should be considered a relatively constrained device, running the FEAC and UCS components directly on it, might impose a considerable overhead on the smart-TV and might result in poor performance. To this end, we consider the possibility of outsourcing these functionalities to other nodes of the Smart Home network. In our testbed implementation, we simulated the smart-TV with an NVIDIA Shield where we implemented a simple live TV streaming application written for Android TV. The application is able to manage the live video streaming allowing the suspend and the resume video actions, and it extracts the video information rating directly from the content metadata when available (natives in Android TVs), or querying external Internet Movie Database (IMDb). The PEP is embedded in the simulated smart-TV as an Android application. It can communicates with the live TV streaming application allowing the enforcement on the streaming. The last component embedded in the smart-TV is the video capture, implemented as Android application as well. It exploits an external camera to capture the live video streaming in front of the TV with frame resolution 650×650, and forwards it to the AI Engine server, which provides a secure endpoint for the transmission.

In our implementation the UCS is installed directly in the (simulated) smart-TV, still it is possible to envision different configurations with the UCS running on a separate device. The UCS itself does not require high performance devices; it is a service that can be installed in any device given that such device can be customized at a certain level. Android devices, as the one used in our testbed, make a perfect fit since they allow installing third party applications. Another possible set-up envisions the UCS as placed in a remote device (e.g., a server) and only the PEP installed in the controlled device. Again, a certain level of customization is required, which makes devices where it is possible to install third-party app perfect candidates. More closed devices might require a deeper modification to deploy the PEP and/or the UCS, such as installing kernel module, or modifying the device firmware. This approach is seldom taken, as it requires a deployment update every time the kernel/firmware version is updated. If these modifications are not possible, a further approach consists of in-lining called external primitives to hijack the operation and enforce UCON policies, or installing the controlling component on a connected device that interacts in a read/write manner with the device that cannot be modified.

3.6.3 | Policy enforcement

The policy enforcement is performed by the PEP, which is embedded in the smart-TV. In our simulated scenario, the PEP is installed into the smart-TV and it interacts with the live TV streaming application to enforce the suspend and the resume actions through a REST API provided by the streaming application. The IFTTT server is implemented using Webhooks [3], that allows to create an HTTP endpoints (hooks) that can be used as a Trigger of the Applet. The *Action* is a service of the IFTTT platform that sends a notification to the smartphone of the TV owner exploiting the IFTTT app. Figure 12 shows the Trigger part which provides a box for specifying the unique Event Name of the Webhooks service.

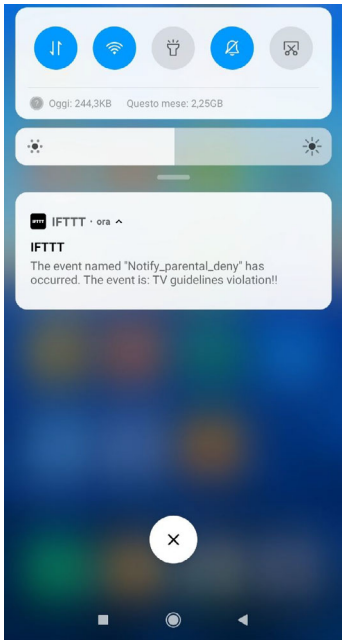
The action can be invoked through an HTTP request to the webhooks service specifying the event name, the service key and the values to transmit in the payload. Such web request is called by the PEP when it extracts the appropriate obligation from the policy with the values to transmit them on the payload (policy violation message or lighting control). The web request is as follows:

$$\boxed{https\ :\ //maker.ifttt.com/trigger/A/with/key/B, A \leftarrow EventName, B \leftarrow Key}. \quad (1)$$

Figure 13 shows the IFTTT notification received on a smartphone.

[3] https://www.ifttt.com/maker_webhooks

FIGURE 13 IFTTT notification



4 | RESULTS

This section presents a set of experiments to evaluate the performance overhead introduced by the proposed framework and to demonstrate its viability.

4.1 | Classification results

In this section are reported the classification results, in terms of accuracy of face detection and age estimation obtained evaluating the FEAC component on the public datasets used in the training phase.

4.1.1 | Face detection and age estimation results

The FEAC component has been tested using the 10% of the total images contained in the Audience Db dataset (testing set) to evaluate both the accuracy in detecting the face and the accuracy in estimating the ages range. As usual, we define:

- **True positive**: The number of correctly classified items belonging to each class;
- **False negative**: For each class, the number of items belonging to that class that have been wrongly classified as belonging to another class;
- **False positive**: For each class, the number of items belonging to a different class, classified by mistake as belonging to that class.

Table 2 concerns the face detection, and it shows the results computed on the testing set. In particular for each age range are reported: (i) the number of faces contained in the testing set (#Faces), (ii) the number of faces correctly detected (true positive), (iii) the total number of faces undetected (false negative), (iv) the number of nonfaces objects detected as faces (false Positive), and (v) the face detection rate, computed as the true positive over the number of faces in the testing set. The total face detection rate reached is 97.21% with a low level of false-positive and negative. The majority of errors are related to the classes with few training samples [48–63] and [60–100]. Table 3 concerns the age estimation, and for each age range are reported: (i) the number of faces detected correctly by the classifier (#Faces detected), (ii) the number of faces with a correct estimation of the age range (true positive), (iii) the number of faces belonging to a specific age range that have been wrongly classified as belonging to another age range (false negative), (iv) the number of faces belonging to another age range that have been wrongly classified as belonging to that age range (false positive), and (v) the age estimation rate for each age range computed as the true positive over the number of faces detected. The overall age estimation rate obtained in the performed experiments is 95.05%. The majority of errors fall in the [8–12] category reaching 85% of age estimation rate. However, it is worth noting that adolescents in this age range might be difficult to categorize even for a human eye, due to different

TABLE 2 Face detection on Audience Db

	[0–2]	[4–6]	[8–12]	[15–20]	[25–32]	[38–43]	[48–63]	[60–100]	Total
#Faces	219	158	129	86	326	147	59	60	1184
True positive	215	157	125	80	320	144	54	55	1151
False negative	4	1	4	6	6	3	5	5	34
False positive	2	3	4	0	5	0	0	3	17
Face detection rate	98.17%	99.36%	96.89%	93.02%	98.15%	97.95%	91.52%	91.66%	97.21%

TABLE 3 Age estimation on Audience Db trained on [0–14] [15–100] age ranges

	[0–2]	[4–6]	[8–12]	[15–20]	[25–32]	[38–43]	[48–63]	[60–100]	Total
#Faces detected	215	157	123	81	315	147	59	58	1152
True positive	210	136	111	70	314	146	59	58	1095
False negative	5	21	18	12	1	1	0	0	57
False positive	6	8	18	5	1	0	0	0	38
Age estimation rate	97.67%	90.24%	85%	86.41%	99.68%	99.31%	100%	100%	95.05%

growth speed. Moreover, it is worth noting that the problem that has been considered in this set of experiments is more complex than the one related to the enforcement of the majority of parental control policies. As a matter of fact, in our experiments, we will consider the ability to discern people with more than 14 years old from people with less than 14 years old.

4.2 | Simulation scenario results

The simulating experiment has been used to evaluate the capability of the framework to estimate the age of spectators from a video stream, simulating what the smart-TV camera is continuously monitoring. To this aim, we simulate the spectators in front of a smart-TV using a video portraying a family in front of a webcam. The video is composed of (i) shares in which the whole family is in front of the camera, (ii) shares with the child alone, and (iii) shares with parents only. The whole video has been reproduced three times, supposing that the spectators will be watching at first content with classification Y, afterward a content with classification PG, and finally one with classification MA. The policies are shown in Section 3.3 have been applied. The duration of the video is 5 min and 3 s with a resolution of 1280×720 at 25 fps. Setting an interval extraction frames of 5 s, 60 frames have been extracted, of which 24 portraying only adults, 11 representing the child alone, and 25 showing the adults and the child together. Out of 115 faces globally present in the 60 frames, the FEAC component was able to correctly detect 113 faces, thus achieving a detection rate of 98.26%. As we can see from Figure 15 the faces undetected, surrounded by red boxes, are peculiar cases such as face cut, people not looking in the direction of the TV, or people too far from the camera. The FEAC was able to correctly estimate the age range of 103 out of 113 faces, thus achieving an age estimation rate of 91.15%. In particular, considering a single frame, 7 times an adult face (out of 74) has been classified as a child and only 2 times the child has been recognized as an adult. Thus the impact on the user experience would have been minimal since the TV program would have only been suspended one time for a few seconds, during the reproduction, according to the policy. Figure 14 shows the confusion matrix of the age estimation process produced by the FEAC component.

4.3 | Real scenario results

A real scenario has been reproduced in a house setting. One of the captured frames is reported in Figure 16, portraying a father with his daughter in front of the TV. The frames have been collected at the same rate of the simulated use case with a resolution 650×650. The frames which have been collected are the following: three frames with only one adult, 12 frames with only one child, and 13 frames with the adult and the child. The FEAC was able to detect 41 faces over 41, thus achieving an accuracy of 100%, whereas it was able to correctly classify 39 over 41 ages with 95% of age estimation rate, with the adult misclassified two times and the child only once.

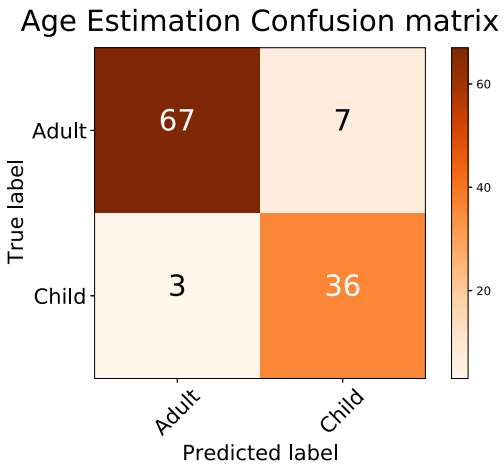


FIGURE 14 Age estimation confusion matrix

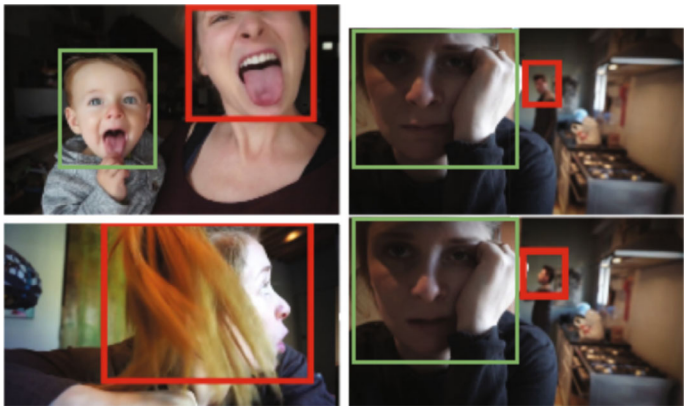


FIGURE 15 Faces misclassified by the FEAC

FIGURE 16 Real experiments frame

4.4 | Result comparison

This section summarizes the results obtained in face detection and age estimation. In particular, is shown the comparison among the approach using a single deep learning network (FEAC component) and the deep learning pipeline presented in Reference 8 (FEL + ACL). Table 4 reports the face detection rate, the age estimation rate, and the computation time of the networks both in the simulation and in the real scenario. The FEAC component that exploits the implementation with a single deep learning network based on Yolo v3, shows an improvement of detection and estimation rate of 2%–3% respect to the FEL and ACL components based respectively on the CNN MMOD and the Deep EXpectation architectures. The proposal to use a single deep learning network reduces the overhead cost in computation time. In fact, we obtained a time computation reduction of 32 ms in the real scenario (frame size 650×650) and 38 ms in the simulation scenario (frame size 1280×720).

4.5 | Performance evaluation

This section evaluates the performance of our framework by monitoring the timing actions of each component. Considering the configuration described in Section 3.6 We measured the timings related to:

- UCON actions
- Face detection and age estimation.

4.5.1 | UCON performances

To perform an extensive evaluation of the performances of the UCON framework, we tested the system starting from a simple case with one attribute until a case with 40 attributes (we recall that in the reference scenario we used only four attributes). We repeated each experiment five times and we report the average. As shown in Table 5, the number of attributes increases the time that the UCS needs to evaluate each request. The *SuspendAccess* and *ResumeAccess* have a similar trend. In fact these actions have a similar execution flow: they perform the policy evaluation and, respectively, they perform the operations to suspend or to resume the session.

4.5.2 | Face detection and age estimation

The time required to extract faces and estimate ages is reported for three different cases: (i) the time to extract faces from a frame on the simulated testbed (1280×720 resolution frame), (ii) the time to extract faces from a frame on the real use case (650×650 resolution frame), and (iii) the time to

TABLE 4 Comparison among single network and pipeline approach

Metric/network	FEAC	FEL + ACL
Face detection rate simulation scenario	98.26%	96%
Age estimation rate simulation scenario	91.15%	89%
Face detection rate real scenario	100%	100%
Age estimation rate real scenario	95%	92%
Computation time real scenario	40 ms	72 ms
Computation time simulation scenario	85 ms	123 ms

TABLE 5 UCON actions performances (execution times in ms)

UCON action/attribute no.	1	4	5	10	15	20	25	30	35	40
UCON TryAccess	312.6	375.8	387.8	389.6	358	380	393.6	433.4	476.2	477.8
UCON StartAccess	79	87.4	94.2	76.8	98.2	90.6	127	106.2	131.4	169.8
UCON RevokeAccess	52	68.3	70.2	72.8	92.6	103.8	121.4	115.6	170.4	146.4
UCON SuspendAccess	36.3	40.5	42.4	49.6	69.8	80.2	94.5	101.2	114.3	120.8
UCON ResumeAccess	74.8	81.5	84.3	90.1	99.6	116.7	135.5	151.1	178.2	190.8

TABLE 6 FEAC performances

Scenario	Time (ms)
T simulation FEAC	85
T real time FEAC	40
T multiple faces FEAC	40

TABLE 7 UCON framework comparison

Framework type	SRT time (ms)	RRT time (ms)
Extended UCON	143.5	196.5
Traditional UCON	171.3	505.8

extract faces from a frame contained a number of faces in the interval [5–30] (650 × 650 resolution frame). Table 6 shows the described performances computed executing the FEAC component on a GeForce GTX1080 graphic card. The results shown as the time needed to extract faces and estimate ages from a frame is extremely dependent from the image resolution and the number of faces in the frame. In the simulation scenario we consider as resolution 1280 × 720 that is higher respect to the resolution in the other two cases. Such aspect, increase the time to extract the faces but can increase the accuracy in the detection and the estimation.

4.5.3 | Framework time

Considering the performances evaluation described in the previous paragraphs, we can estimate the total response time of the framework taking into account our reference scenario. In case of policy violation, we are interested to minimize the time that a specific not permitted action can be performed. Thus considering the TV parental guideline violation (case in which a child remains alone during the streaming of a PG program), we can estimate the reaction time as the overall suspend response time of the proposed framework, $SRT_{ExtendedFramework}$, as reported in Formula (2).

$$\boxed{SRT_{ExtendedFramework} = T_{FEAC} + T_{UCSSuspendAccess} + T_{StopStreaming}}. \tag{2}$$

Considering the real time scenario, the suspend response time of the framework is the sum of the AI engine to detect and estimate the ages (40 ms), the time for policy reevaluation to decide the suspension of the resource access (40.5 ms), and the time that the PEP needs to perform the stop enforcement on the video streaming (evaluated in 63 ms). Thus, the overall amount of time necessary to stop the reproduction of a video in case of policy not respected in a real scenario is 143.5 ms.

As well as the evaluation of the suspension time, is interesting analyze the resume reproduction time (RRT), that is, the time necessary to resume the reproduction of the video when the policy is once again matched. Such time is expressed in Formula (3) and it is composed by the AI Engine time to detect the people in front of the camera and their ages (40 ms), the time for policy reevaluation to decide to resume the access to the resource (81.5 ms) and the time that the PEP employs to resume the streaming (evaluated in 75 ms). Thus the overall amount of time necessary to resume the reproduction of a video due to the attribute changing is 196.5 ms.

$$\boxed{RRT_{ExtendedFramework} = T_{FEAC} + T_{UCSResumeAccess} + T_{ResumeStreaming}}. \tag{3}$$

Considering the same operations (stop and resume video execution) using the traditional UCON model, the suspend and resume access would have been replaced by the RevokeAccess in case of policy violation, and a TryAccess in case of resume conditions. In addition, has to be considered the time to start the video (90 ms). The two time are reported in Formulas (4) and (5).

$$\boxed{RRT_{OriginalFramework} = T_{FEAC} + T_{UCSTryAccess} + T_{StartStreaming}}, \tag{4}$$

$$\boxed{SRT_{OriginalFramework} = T_{FEAC} + T_{UCSRevokeAccess} + T_{StopStreaming}}. \tag{5}$$

The estimation time computed considering four attributes is 505.8 and 171.3 ms, respectively, for the $RRT_{OriginalFramework}$ and $SRT_{OriginalFramework}$. Such evaluation shows the benefits of the extended UCON framework that, through the introduction of the suspend and resume access, allows to decrease the response time as shown in Table 7.

5 | RELATED WORK

An application of UCON in IoT environments has been presented in Reference 3. The authors propose a distributed model of the standard UCON framework, discussing a Smart Home use case. An implementation specific for IoT communication protocols is presented in References 2,20, where the authors present the integration of the UCON paradigm in the MQTT and CoAP workflow. The focus of these works, however, is not centered on obligations, they only exploit in an appropriate way the authorization constructs. On the side of IFTTT an effort to create simple policy algorithms for IoT is presented in Reference 21. This work focuses more on the policy specification and how it could be predicted according to the user features and the specific IoT domain. This work, though, does not study the way policies are written, or the obligation format and context but focuses more on the part of how to create a policy according to what is happening in a specific IoT domain. Furthermore, enforcement mechanisms are not considered in their analysis. In Reference 6, the author describes the development of a specific framework that can modify the concepts of the IFTTT platform so as to provide services that can be used in order to make home automation systems more secure. This work, though, creates specific services in the IFTTT platform and also depends only on the security mechanisms of IFTTT without any control on the installed environment and, also, no continuous way of evaluation or policy enforcement. In Reference 7, the authors describe also some possible security and privacy risks in the *Applets* of IFTTT.

Concerning the XACML model for policies, in Reference 22, the authors present a description of obligation expression based on examples in grid and networking security. The work is very specific to this limited environments and does not provide a way to formalize the obligations semantic. Finally, in Reference 23, the authors present a variation of XACML specific for obligation extraction. First, they propose a method of creating a new file for obligations in XACML which is produced by the initial file but they do not give a clear description of how the obligation part should be constructed so as to be easily readable by both the PDP and the PEP. Though interesting, their approach is not based on standard XACML, differently from our solution that integrates in XACML 3.0 policies without requiring any modification to the standard architecture and workflow. Usage control for data management has been used for controlling dynamically the right to use files on mobile devices and in the cloud, respectively in References 1,24. These works are mainly focused on the description of the Usage Control framework infrastructure, presenting generic policies. The present work describes instead a real application of the Usage Control in a real environment, with effects not only on security, but safety as well.

A framework which exploits the embedded camera in a smart-TV for user authentication is presented in Reference 25. By authenticating users the framework also performs parental control enforcement, however, differently from Smart Parental Advisory the age recognition is based on the manual assignment of the age to the recognized face. On the other hand, our framework performs age recognition automatically, without exploiting the identity. Hence, smart parental advisory is able to recognize new adults and children without the need of retraining it for any new spectator, being thus more general. A similar machinery to the one of Reference 25 has been patented for commercial use in Reference 26. Deep learning for face recognition has been used with high accuracy in Reference 27. However, this work only focuses on classification aspects, without considering parental control aspects. A recent work focused instead only on age estimation is the one presented in Reference 28, where deep learning algorithms have been used to automatically infer the age of people from face pictures. In our work instead the process of age recognition is more challenging since the faces on which the age is estimated are extracted from live streams. Another effort toward exploiting smart device capabilities is shown in Reference 29, where the authors focusing on extending parental control capabilities through rooting on smart-TVs. This approach though needs specifically modified smart-TVs and software installed to them, that may cause instability of the operating system and break of the warranty. Instead our approach does not require such modifications to smart-TVs. Finally, in Reference 30, the authors suggest a blockchain framework that monitors the actions over resources in a Smart Home case, rewarding good behaviors and penalizing bad ones. But, the authors do not implement any verification mechanisms on the tasks needed to be fulfilled. They only suggest the usage of surveillance cameras for such a task. In our work, though, we present a face detection and age estimation algorithm, that is able to identify the presence or not of kids and adults when they are in front of a camera.

6 | CONCLUSION

Controlling access rights to operations and devices in Smart Home settings enables the definition of security, safety and management policies able to improve the level of customization and quality of provided services. In this article we have presented an architecture exploiting UCON and IFTTT to manage safety and security in a Smart Home setting, focusing on a parental control use case. The presented use case strongly motivate our work, showing the enforcement of safety policies in an interconnected Smart Home environment, which involves three cooperating and interconnected devices. The proposed approach and the novel introduced models show improvement for what concerns policy expressiveness and overall system effectiveness and performance. By exploiting the IFTTT webservice, this work also provides a method to better exploit the UCON obligations, by using an actual protocol with standardized expressions.

As future extensions, we plan to design more structured IFTTT-based obligations with dedicated tags possibly to be proposed as standard extensions. Furthermore, we plan the design and testing of more complex use cases, with devices having a dedicated UCS and IFTTT capable applications to enforce their own and third parties obligations.

ACKNOWLEDGMENTS

The activities leading to this research have been partially funded by the EU H2020 project SPARTA GA 830892 and the EU H2020 project SIFIS-Home GA952652.

ORCID

Giacomo Giorgi https://orcid.org/0000-0001-8037-1386
Andrea Saracino https://orcid.org/0000-0001-8149-9322

REFERENCES

1. Carniani E, D'Arenzo D, Lazouski A, Martinelli F, Mori P. Usage control on cloud systems. *Futur Gener Comput Syst*. 2016;63:37-55. https://doi.org/10.1016/j.future.2016.04.010.
2. La Marra A, Martinelli F, Mori P, Rizos A, Saracino A. Improving MQTT by inclusion of usage control. In: Wang G, Atiquzzaman M, Yan Z, Choo KR, eds. *Security, Privacy, and Anonymity in Computation, Communication, and Storage - 10th International Conference, SpaCCS 2017; December 12-15, 2017, Proceedings. 10656 of Lecture Notes in Computer Science*. Guangzhou, China: Springer; 2017:545-560.
3. La Marra A, Martinelli F, Mori P, Saracino A. *Implementing Usage Control in Internet of Things: A Smart Home Use Case, Trustcom/BigDataSE/ICESS*. Sydney, Australia: IEEE Computer Society; 2017, 2017:1056-1063.
4. Park J, Zhang X, Sandhu RS. Attribute mutability in usage control. In: Farkas C, Samarati P, eds. *Research Directions in Data and Applications Security XVIII, IFIP TC11/WG 11.3 Eighteenth Annual Conference on Data and Applications Security, July 25-28, 2004. 144 of IFIP*. Sitges, Catalonia, Spain: Kluwer/Springer; 2004:15-29.
5. Park J, Sandhu RS. The $UCON_{ABC}$ usage control model. *ACM Trans Inf Syst Secur*. 2004;7(1):128-174. https://doi.org/10.1145/984334.984339.
6. Vorapojpisut S. A lightweight framework of home automation systems based on the IFTTT model. *JSW*. 2015;10(12):1343-1350. https://doi.org/10.17706/jsw.10.12.1343-1350.
7. Surbatovich M, Aljuraidan J, Bauer L, Das A, Jia L. Some recipes can do more than spoil your appetite: analyzing the security and privacy risks of IFTTT recipes. In: Barrett R, Cummings R, Agichtein E, Gabrilovich E, eds. *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, April 3-7, 2017*. Perth, Australia: ACM; 2017:1501-1510.
8. Giorgi G, La Marra A, Martinelli F, Mori P, Saracino A. Smart parental advisory: a usage control and deep learning-based framework for dynamic parental control on smart TV. In: Livraga G, Mitchell CJ, eds. *Security and Trust Management - Proceedings of the 13th International Workshop, STM 2017, September 14-15, 2017. 10547 of Lecture Notes in Computer Science*. Oslo, Norway: Springer; 2017:118-133.
9. La Marra A, Martinelli F, Mori P, Rizos A, Saracino A. Using IFTTT to Express and Enforce UCON Obligations. In: Heng S, López J, eds. *Information Security Practice and Experience – Proceedings of the 15th International Conference, ISPEC 2019, Kuala Lumpur, Malaysia, November 26-28, 2019, 11879 of Lecture Notes in Computer Science*. Kuala Lumpur, Malaysia: Springer; 2019:213-231.
10. Pretschner A, Hilty M, Basin DA. Distributed usage control. *Commun ACM*. 2006;49(9):39-44. https://doi.org/10.1145/1151030.1151053.
11. OASIS eXtensible access control markup language (XACML) Version 3.0; 2013.
12. Zhang X, Parisi-Presicce F, Sandhu RS, Park J. Formal model and policy specification of usage control. *ACM Trans Inf Syst Secur*. 2005;8(4):351-387. https://doi.org/10.1145/1108906.1108908.
13. Colombo M, Lazouski A, Martinelli F, Mori P. A proposal on enhancing XACML with continuous usage control features. In: Desprez F, Getov V, Priol T, Yahyapour R, eds. *Grids, P2P and Services Computing Proceedings of the CoreGRID ERCIM Working Group Workshop on Grids, P2P and Service Computing, 24 August*. Vol 2009. Delft, The Netherlands: Springer; 2009:133-146.
14. Martinelli F, Mori P, Saracino A. Enhancing android permission through usage control: a BYOD use-case. In: Ossowski S, ed. *Proceedings of the 31st Annual ACM Symposium on Applied Computing, April 4-8, 2016*. Pisa, Italy: ACM; 2016:2049-2056.
15. Redmon J, Farhadi A. Yolov3: an incremental improvement; 2018. arXiv preprint arXiv:1804.02767.
16. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: a large-scale hierarchical image database. Paper presented at: Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition. Miami, FL; 2009:248-255.
17. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning; 2017.
18. Rothe R, Timofte R, Gool LV. *DEX: Deep EXpectation of Apparent Age from a Single Image*. Santiago, Chile: IEEE International Conference on Computer Vision Workshop (ICCVW); 2015:252-257.
19. Eidinger E, Enbar R, Hassner T. Age and gender estimation of unfiltered faces. *IEEE Trans Inf Forens Secur*. 2014;9(12):2170-2179.
20. Rizos A, Bastos D, Saracino A, Martinelli F. Distributed UCON in CoAP and MQTT protocols. In: Katsikas SK, Cuppens F, Cuppens N, et al., eds. *Computer Security - ESORICS 2019 International Workshops, CyberICPS, SECPRE, SPOSE, and ADIoT, Luxembourg City, September 26-27, 2019 Revised Selected Papers. 11980 of Lecture Notes in Computer Science*. Luxembourg: Springer; 2019:35-52.
21. Nadkarni A, Enck W, Jha S, Staddon J. Policy by example: an approach for security policy specification. *CoRR*. 2017;abs/1707.03967.
22. Demchenko Y, Koeroo O, Laat DC, Sagehaug H. Extending XACML authorisation model to support policy obligations handling in distributed application. In: Schulze B, Fox GC, eds. *Proceedings of the 6th International Workshop on Middleware for Grid Computing (MGC 2008), held at the ACM/IFIP/USENIX 9th International Middleware Conference, December 1-5*. ACM: Leuven, Belgium; 2008:5.
23. Chadwick DLM. Obligation standardization; 2009:1-5.
24. Lazouski A, Martinelli F, Mori P, Saracino A. Stateful data usage control for android mobile devices. *Int J Inf Sec*. 2017;16(4):345-369. https://doi.org/10.1007/s10207-016-0336-y.
25. Lee S, Sohn M, Kim D, Kim B, Kim H. Smart TV interaction system using face and hand gesture recognition. Paper presented at: Proceedings of the 2013 IEEE International Conference on Consumer Electronics. Berlin, Germany; 2013:173-174

26. Kwoh DS, Mankovitz RJ. Apparatus and method for total parental control of television use. U.S. Patent No. 5,382,983. 1995. http://www.freepatentsonline.com/5382983.html.
27. Taigman Y, Yang M, Ranzato M, Wolf L. *DeepFace: Closing the Gap to Human-Level Performance in Face Verification*. Columbus, OH: IEEE Conference on Computer Vision and Pattern Recognition; 2014:1701-1708.
28. Dong Y, Liu Y, Lian S. Automatic age estimation based on deep learning algorithm. *Neurocomputing*. 2016;187:4-10. https://doi.org/10.1016/j.neucom.2015.09.115.
29. Majchrowicz M, Kapusta P, Faustryjak D, Jackowska-Strumillo L. System for remote parental control and management of rooted smart TVs. Paper presented at: Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW). Swinoujscie, Poland; 2018:357-360.
30. Suchaad SA, Mashiko K, Ismail NB, Abidin MHZ. Blockchain use in home automation for children incentives in parental control. *MLMI'2018*. New York, NY: Association for Computing Machinery; 2018:50-53.

How to cite this article: Giorgi G, La Marra A, Martinelli F, Mori P, Rizos A, Saracino A. Exploiting If This Then That and Usage Control obligations for Smart Home security and management. *Concurrency Computat Pract Exper*. 2021;e6189. https://doi.org/10.1002/cpe.6189