Understanding User Needs in Smart Homes and How to Fulfil Them

Andrea Mattioli[1,2][0000-0001-6766-7916] and Fabio Paternò[1][0000-0001-8355-6909]

[1] CNR-ISTI, HIIS Laboratory, Pisa, Italy
[2] Department of Information Engineering, University of Pisa, Pisa, Italy
{andrea.mattioli,fabio.paterno}@isti.cnr.it

Abstract. Smart homes are becoming a widespread reality given the increasingly available number of connected objects and sensors. However, it is still unclear what people expect from automations that are made possible by this technological evolution. In addition, it is unclear whether current trigger-action programming (TAP) languages offer sufficient operators and constructs to specify the desired automations. In this paper, we report on a study aiming to provide useful elements to address such issues. It involved 34 users without experience in IoT programming who created 204 desired home automations. We discuss an analysis of such results in terms of the relationships found between smart-home components and of the requirements for novel operators in TAP languages.

Keywords: Smart Homes, End-User Development, Trigger-action Programming, User Requirements.

1 Introduction

Personalizing smart homes, which are spaces where objects and devices capable of connecting to the Internet are often used in conjunction with online services, has recently gained popularity. Over the last few years, the widespread of these objects has led to the growth of the Internet of Things (IoT) vision. It is a pervasive technology that according to Statista [30] will continue to expand in the next few years. A relevant approach to capitalize on the new possibilities that this landscape offers is Trigger-Action Programming (TAP), which is an End-User Development (EUD) approach that allows people who are not experts in programming to generate custom automations to reach their goal using the rule metaphor. TAP has shown to be an effective approach [5, 32] to configuring automations including objects, devices and services that behave in a concerted manner. A TAP rule includes a "trigger part", that can be formulated in terms of events and conditions, and an "action" part, defining what will be activated at the occurrence of the trigger part. Although there is evidence that TAP is understandable by most end users, there are still problems with the EUD platforms for personalization [12]. In general, understanding how users interpret and use automations requires more research [5, 9, 28]. It is currently unclear, for example, how aware users are of the advantages and the hazards of these platforms. For instance,

users can create automation with problems deriving from misunderstandings of the temporal relations of triggers and actions [5] and from the complex interactions between rules [36]. They can also inadvertently create automations harmful to their privacy and to the security of their environments [6]. It is hence crucial how to assist them in better orchestrating behaviours involving more objects and automation rules. Analyzing what potential users anticipate from home automation systems in terms of rule functionalities, constructs and operators can be a first step to designing systems and languages more capable of matching users' mental models, and ultimately lead to the clarification of these issues.

2 Related Work

2.1 TAP Rules and Extensions

Different studies investigated the possibility of expanding TAP syntax with further functionalities introducing more flexible rule formulations. These efforts focused on various aspects, for instance, introducing contextual information such as "when" and "where" [14, 17], designing the possibility of using the fuzziness concept and other space and time aspects in trigger and actions specification [4], explicating the "not" and "revert" operators [25], introducing new operators such as time counters, accumulators and rule chaining [28], or allowing to define automations that refer to more or less abstract levels of abstraction [10]. One of the goals of these efforts is to allow users to generate automations that fit their needs more precisely. At the same time, it is crucial to consider how to balance the expressivity of the rule specification language with its easiness of use [32]. More expressivity can hinder the adoption of the platform because it can become more difficult to understand and use it, but there is still the need to be able to express the desired behaviours. Hence, one relevant aspect is understanding what functionalities users expect, and then it is useful to analyse the more spontaneous natural language descriptions of their intents to derive the necessary operators and constructs that a language for trigger-action rules should support. Indeed, such languages are evolving in order to more flexibly support users' needs. For example, IFTTT started as a language supporting only single trigger/single action rules and recently has introduced in a professional version the possibility of multiple actions and filters to select the actions to perform through specific scripts.

2.2 Eliciting users' preferences

Prior work in characterizing users' behaviour within smart home systems followed different approaches, ranging from online surveys to extended experiments that require a full smart home installation. A way to directly elicit users' preferences from the products of their interaction with automation personalization systems is the crawling and then analysis of publicly available automation rules [26, 32, 33, 34]. Another approach involves potential users through online surveys, probing their expectations about home automation systems. Examples of this approach are in the first part of the study in [32], where the authors collected five desired smart-home behaviours from

respondents to the survey. Then, they analysed if it would be possible to implement these desires with trigger-action programming and whether it would require multiple triggers or actions. In one contribution [27] the authors exploited an online survey to investigate users' desires for smart devices and features at home, focusing on situations where participants wished for a more intelligent device or service. In another one [29] the authors collected home automation scenarios using a survey. They provided participants with a fixed house model and devices list, intending to gather as many possible diverse smart home scenarios while maintaining a level of plausibility with real-world installations. An approach based on a more ecological setting can be providing users with a way to describe and collect automation rules at their homes during a longer period [23]. In a study [12] the authors devised a pen-and-paper kit to allow participants to note down a possible automation whether it comes to their mind during their daily activities. After a first home appointment, participants had one week to collect automations in the given structure, which permits the definition of rules with one trigger and one action, eventually enriched by contextual attributes such as "where", "who", "when", and "which". Brich and colleagues [7] conducted a one-week study where participants had to define automations using two notation kits, one rule-based and the other process-based. The study started with a tour, where participants were introduced to the concept of automation and could start to formulate use cases in a free-form manner. Then, participants were acquainted with the kits, and during the week they could come up with more automation ideas. The main goal of the inquiry was to assess the benefits and drawbacks of the two notations. In another study [28] the authors ideated, implemented and installed in the participants' houses a smart-home toolkit, and organized creativity workshops with the recruited families to facilitate the ideation and writing of automations. The families then configured and used the system for six weeks. A crucial aspect emerging from this research is that social aspects are not considered enough in current home automation systems.

2.3 Research Objectives Definition

Some common traits can be drawn from an analysis of the literature. Different studies aimed at eliciting users' preferences or patterns in device uses are based on a posteriori analysis of automation datasets. Other studies surveyed how potential users describe automations. However, only a few have involved participants with indications aiming at eliciting the desired automations, at times using a guiding template, such as an environment and devices list, or a rule structure, allowing them to write rules creatively and freely in their spaces. Also, little attention has been devoted to investigating the final objective (such as comfort, well-being, security, or energy saving) of the users' automations composition, and the relations between the immediate (such as illumination in a room) and the long-term objectives. Another aspect that previous work focused on is defining extensions to the TAP operators and structure, by designing and assessing interfaces (and the underlying systems) for this goal. Nonetheless, few prior works aimed at understanding how people think and use rule operators, values, and connectors between rule parts. A way to inquire about these aspects can be letting participants compose automations through a template that allows the defini-

tion of both simple and advanced automations (e.g., including multiple triggers and actions), also letting them define the notation elements for formalizing the operators and the relations between rule elements in a way most suitable for them. To contribute to addressing such issues, we defined the following research objectives to drive our study:

- What functionalities do potential users expect from automations in a smart home setting?
- What TAP constructs and operators are necessary to specify these desired behaviours?

3 The User Study

Our goal was to investigate which functionalities would people who are not programming experts expect in a home automation system, and to find whether any relationships emerge between these functionalities, or between these and users' long-term preferences. Another goal was to understand how they would define these desired behaviours, for instance, which operators they would use, or if there are any specific rule constructs that participants use for describing them. As described in the previous section, some existing work has been concerned with eliciting the personalization features that people expect or desire in smart home environments. A small number of these [7, 12] have developed kits to allow participants to define automations during their daily activities. However, to our knowledge, there is a lack of a study where participants could describe automations using a syntax that is both expressive and realistic (implementing Event-Condition-Action rules) and extensible according to their needs (e.g., defining new operators). Furthermore, there is no study where the automations are also analysed from a linguistic point of view, where the descriptions in natural language can be compared with their formalized versions. For these reasons, we carried out a one-week user study where we collected automation rules from participants in a format that allowed us to subsequently analyse them.

3.1 Tasks

The tasks that participants were asked to carry out were first to compile a list of automations that they imagine could be useful in their living place, expressing these automations using a natural language description. After that, they also had to formalize them using a provided rule template. To address the first research objective, participants could imagine having any IoT device installed in their homes and could integrate them using web services or apps. Hence, they were not forced to use a predefined list of devices and services, or only technology that they know is already available commercially, but they could include technologies that they expected to be available currently or shortly. We adopted this approach to balance the plausibility of the gathered automations with their capability of matching the participants' needs. Participants were given one week to complete the automation list.

3.2 Rule Template

To allow participants to define automations in a non-ambiguous manner, but at the same time avoid forcing them to use a specific language, we provided them with a document including a template and some examples of common automations, such as "When the user is in the garden and it's between 19:00 and 05:00, turn on the garden lights" (See table 1 for the corresponding description in the template).

Table 1. An example of automation is described using the template, where each row represents a rule element. The title of this automation is "Garden Light", the goal is "Comfort", and the context is "Spring".

ECA	Environment	Channel	Functionality	Operator	Value	Next Op.
Event	No	Position	User position	Equal	Garden	And
Condi-tion	No	Date and Time	Time	Between	19:00 - 05:00	
Action	Garden	External Light	Light on	Equal	TRUE	

The template's header contains three fields related to the whole rule, namely its title, the long-term goal, and the context of actuation, a more high-level description of the scenario in which the automation could be activated. The template's body comprises seven fields specific to each trigger and action. These fields are the "ECA" class (the only field with fixed values the participants had to choose from, which are Event, Condition or Action); the environment (such as a specific room, valid for the entire house, or the room user is currently in); the channel, which can be a more or less specific description of an object or device (such as "door sensor", "smartphone" or "Alexa"), as well as a service (such as "weather forecast"); the specific functionality of the channel (for instance, "room temperature" or "is raining"); the desired value and operator; and a Next operator (which connects a rule element to the next one). The "Context" and "Next Operator" fields were labelled as optional, as we considered the former as additional information not crucial to the understanding of the rule, and the latter as not always needed, and eventually derivable from the rule elements or from the natural language descriptions. However, participants could leave a field blank if it was not pertinent to the automation, such as the "Environment" field in the first two rows of the example.

This structure is partially inspired by the IFTTT syntax, which is organized into channels (the available device or service), functionalities and then user-defined details of the automation, and on the hierarchical organization of the HomeKit dataset [17]. We adopted the Event-Condition-Action syntax because it allows expressing behaviours with adequate expressiveness for the smart home context [3, 8, 13, 16], and because of its widespread adoption in smart home systems. In the considered template, the event is what causes the system to activate the action part. If one or more conditions are present, these are checked after the event verification to assess whether the state of the environment is satisfactory for the activation of actions. In the case of condition(s) without an event, the beginning of the condition is considered an event.

To gather information about the second research objective, we did not put hard constraints on the syntax. Hence, participants could use as many rule elements as desired, ordering them as they prefer (for instance, adding another "event" check after an action). They could link them freely by defining the "next operator" as they prefer, and also complete the other field in a way that better fits their automation idea. In this way, using a plausible but flexible syntax, we could elicit participants' desired automations and which operators would be necessary to formalize them, without restricting this process to a specific language or user interface.

3.3 Participants

The participants in the user study were recruited from a Digital Humanities degree course. Thirty-four users (eleven males) with ages ranging between 23 and 29 years were involved in the study. During their degree studies, they were exposed to Web Programming courses, but they had no experience in IoT (besides using widespread devices such as Amazon Echo) or trigger-action programming.

During the first encounter, which occurred in person, participants were introduced to the interactions with the Internet of Things and to trigger-action programming. They were presented with key concepts such as the distinction between events and conditions and with some examples of common automations. In a second briefer meeting, the key concepts from the first one were recalled, and then they were instructed about their tasks.

3.4 Collected Data

Participants overall produced 204 automations, comprised of the structured and natural language descriptions of the functionality they wanted to express. The structured rules included overall 735 "rule elements" (every single row in the rule structure table, on average 3.603 elements per rule). The dataset containing the automations generated in both structures has been uploaded on GitHub [20].

Some high-level observations can be made from a first look at the natural language descriptions. Two main strategies were used in describing the automations. Participants used both a direct style, as if they were asking the system to perform the actions ("then activate"; "make the fridge suggest recipes from its contents"; "tell Alexa to do this"), and a more impersonal style, describing the situation in the environment that results when the actions execute ("the watering mechanism is activated"; "the cameras turn on and alarm notifications are activated"). We can also distinguish between automations described using a rule-like style ("if this, then do that") and with a more descriptive style ("assuming the house has an entrance door with a smart handle, make sure that…"). Furthermore, in some cases they described the rule indicating that its triggering directly depends on the user, environment, or device state or action, without the mediation of a sensing layer ("if the user is studying"; "if it's raining"; "when it's 8:00"; "if the kitchen temperature is below 17 degrees"), while in other they specify that the sensor is the subject that is performing the check that may lead to the triggering of the automation ("when the bed sensor detects the user"; "when the

sensor measures soil humidity below 60%"). Participants seldom used specific brands of devices, mostly for voice assistants, cleaning robots and gaming consoles (respectively 14, 3 and 2 occurrences). In some cases, participants used vague descriptions, both for the trigger part ("when the courier is near", "the house is not yet warm and it is cold outside") and the action part ("make the water very cold"). We counted 4 definitions of this kind, while in most cases, they specified a precise value. Other preliminary observations concern how users used the optional context field. Eight users have used this field to further describe their automations. Examples of these descriptions are "every morning", "any morning when an alarm is set", "winter months only", and "term time".

To make data more understandable, we then analyse these automations to define some classes to group them. We defined a new "Use case category" class from a combination of the "Channel" and the "Functionality" columns. This new definition comprised 22 "middle level" categories (higher than the single functionality but more concrete than the final goal), obtained by grouping the most frequent and conceptually near functionalities defined by participants. This classification has been done considering the gathered data, and how channels have been combined in related work [12, 32]. For analysing the "Goal" and "Environment" fields, we instead directly use the user-assigned keywords (1 or at most 2 in the case of "Goal"), merging them when they represent the same concept (e.g. "Marco's room" and "Kid room"). Then, we analysed the relations between categories, goals and environments.

3.5 Limitations

About the limitations of this study, one aspect to note is that participants have similar backgrounds. This could have impacted the variety and the choice of functionalities in the automation rules that they produced. Another thing to consider is that participants may have been influenced by the example automations and by the rule structure provided to them. However, having one week to collect the automations should have allowed them to reflect on their daily situations instead of repeating examples. Also, we asked them to define first the rule in a spontaneous way using natural language, and not using the specific rule language to prevent its structure from affecting the reasoning.

4 Analysis

After having provided an overview of the collected data, in this section we will analyse them considering both the structured rules and the natural language descriptions generated by participants.

4.1 What functionalities do people expect from a smart home system?

Once a categorization for the functionalities was defined, we started to analyse the relations between them. In the case of doubts, we cross-checked the rule structure with its natural language description to disambiguate the behaviour intended by the

participant. Some initial insights can be obtained by considering the frequencies and percentages of the classes in all positions of a rule, in the trigger part, and in the action part (see Table 2).

Table 2. The identified classes with their use in frequency and percentage in the trigger and action part of the rules.

	count (all)	% (all)	count (t)	% (t)	count (a)	% (a)
Feeding	30	4.1	18	4.7	12	3.4
Alarm	11	1.5	3	0.8	8	2.3
Hygiene	42	5.7	18	4.7	24	6.9
Device	48	6.5	17	4.4	31	8.9
Door and window	45	6.1	19	4.9	26	7.4
Temperature	45	6.1	21	5.5	24	6.9
Air and humidity	16	2.2	13	3.4	3	0.9
Gardening	9	1.2	1	0.3	8	2.3
Lights	50	6.8	7	1.8	43	12.3
Notification	81	11	0	0	81	23.1
Systems	31	4.2	9	2.3	22	6.3
User detection	30	4.1	30	7.8	0	0
Smart object	27	3.7	13	3.4	14	4
Personal device	11	1.5	7	1.8	4	1.1
Communication	9	1.2	0	0	9	2.6
Kids	12	1.6	6	1.6	6	1.7
Scheduling	91	12.4	82	21.3	9	2.6
Presence	73	9.9	73	19	0	0
Pets	18	2.4	11	2.9	7	2
Entrances	30	4.1	17	4.4	13	3.7
Data	13	1.8	7	1.8	6	1.7
Weather	13	1.8	13	3.4	0	0

It can be observed that participants defined the trigger parts making wide use of the “presence” and “scheduling” classes, together accounting for more than 40% of all trigger instances. Rules using the presence trigger involved mainly checking if the user is at home, but also in a specific room or position such as near the window, and also to indicate a generic person (e.g. for detecting a presence in the garden). The scheduling trigger has been used both as a condition (limiting the execution of an automation to a specific time) as well as an event (to start the further condition checks or the action part). For the action part, the “notification” functionality and, to a lesser extent, the “light” controls were the most used. An insight into the relations between the classes can be obtained by considering, for each automation, when a class is found together with another and then assessing the phi correlation between them (see Figure

1). A weak positive correlation has been found between the “gardening” and “air and humidity” functionalities (due to the relatively common automations that activate the sprinklers when the humidity of the terrains is under a certain value), and a weak negative correlation between the “notifications” and “lights”. Since these two classes represent the most used actions, we can infer that a good portion of automations gravitates around one or the other of these functionalities.

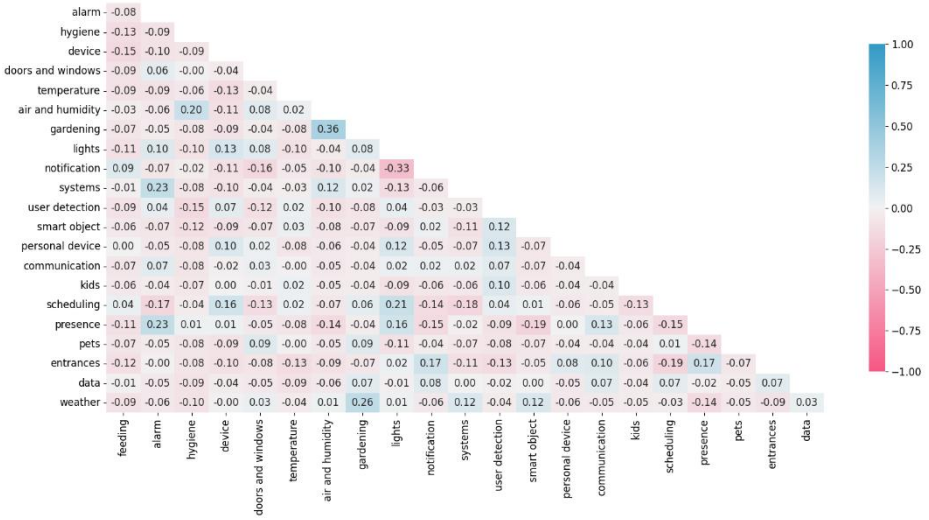


Fig. 1. Phi correlation between the classes, calculated from the frequencies of their joint occurrences in automations.

There are other positive and negative relations that, although do not represent a correlation or are very weak, can give some insight into other common use cases. Some examples are between “alarms” and “systems” (mostly representing safety and security rules, activating alarms and other prevention systems when users leave home or making emergency calls when problems in the house are detected), “air and humidity” and “hygiene” (mostly automations to circulate air in the house, to prevent mould or smells) “scheduling” and “devices” (e.g., activating the robot cleaner or the music player at a specific time, or dimming the light when the ebook reader is in use in the evening), “gardening” and “weather” (often related to not activating sprinklers when it is raining).

To further analyse the automations, we then look for relations between functionality classes and rule goals, and functionalities and environments. Regarding the class–goal relation, considering the absolute number of occurrences (see Figure 2) we can observe the high presence of the “comfort” goal, being the most selected in most of the classes (all except alarms, systems, communication, kids, and entrances). Other connections (total occurrences equals or more than 15) emerge between “presence” with “energy saving” and “safety” rules, “well-being” with “temperature”, and “security” with “presence” and “entrances”.

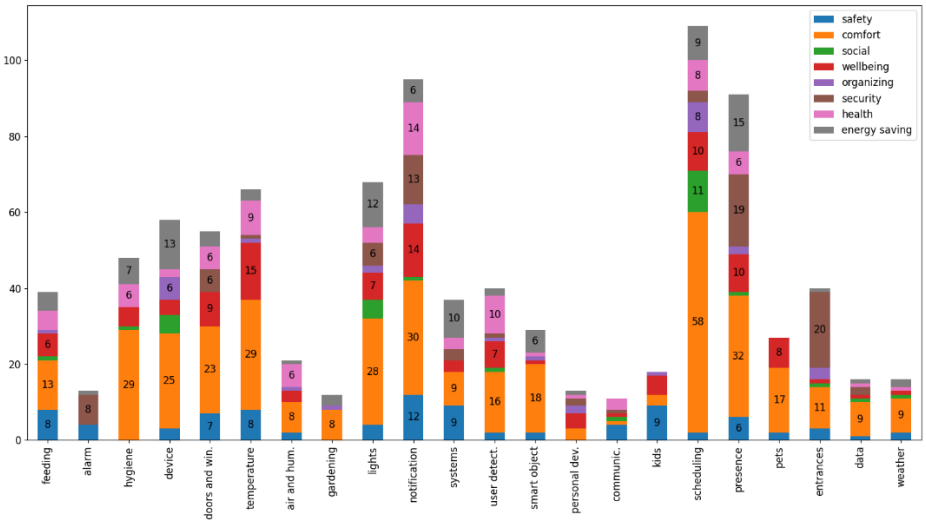


Fig. 2. Frequencies for the goals in each class (only counts > 5 are shown).

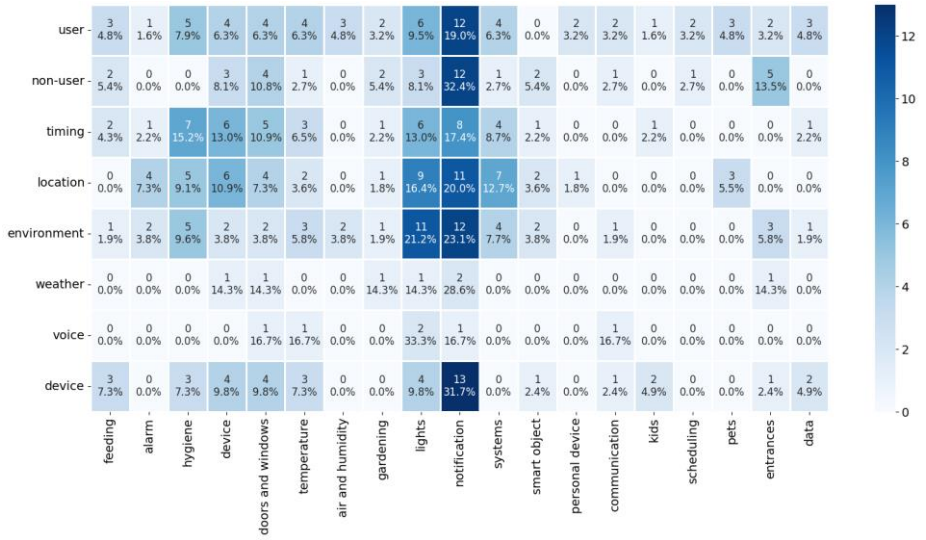


Fig. 3. Relations between the events that cause the triggering (on the y-axis) and the classes of actions (x-axis).

Considering the frequency count, the top-3 most common functionality classes for each goal are respectively scheduling, lights and devices (11 and 5 occurrences) for the social goal; temperature, notifications, scheduling and presence (15, 14, and 10) for wellbeing; scheduling, devices and notifications (8, 6 and 5) for organizing; scheduling, presence and notification (58, 32 and 30) for comfort; notifications, kids

and systems (12, 9 and 9) for safety; notifications, user detections and temperature (14, 10 and 9) for health. Regarding relations between environments and categories, we observed stronger ties between “no specific location” with the “scheduling” and “notification” classes (57 and 50 occurrences respectively). Other relations are in “kitchen” rules with feeding (28 occurrences), “entire home” with “presence” (22) and “doors and windows” (21), “bathroom” rules with “hygiene” (21), “living room” with “devices” (19) and “bedroom” with “scheduling” (16). Another investigated relation is between the events that could trigger the rule and the category of the actions in that rule. From Figure 3, it can be observed that the “notification” action can be found transversally in all triggering classes. This is not the case for the other actions, which are overall less present and more sparse.

4.2 Which TAP operators and structures are necessary to express these behaviours?

Operators for values and between rule elements. Regarding how participants defined the “next operator” field, the “and” operator was by far the most common. It was commonly used to connect both elements in the trigger (170 occurrences) and in the action part (109 occurrences). By comparison, the “or” operator was found only 11 times. Other used operators were the “iteration”/“every X minutes” (to specify how often a trigger check or an action should be repeated), the “while t<T” (to indicate that an action should remain active for a given period), and the “and after X minutes”. The “do” operator was used five times, to indicate when the end of the trigger part and the start of the action, or to signal multiple actions. Also, the “while” operator was used two times to indicate a sustained action [22] temporally linked to a condition, for instance, “while the user is cooking activate the hob extractor fan”. Considering also the natural language descriptions, “While” was overall used six times, all the times with this connotation. Regarding the operator field, the “equal” operator was the most used (577 times). Other commonly used operators were more and less than, more equal and less equal, not, and between. Besides these standard operators, the “for” was used in 3 occurrences to indicate a period associated with conditions (“for 10 minutes”).

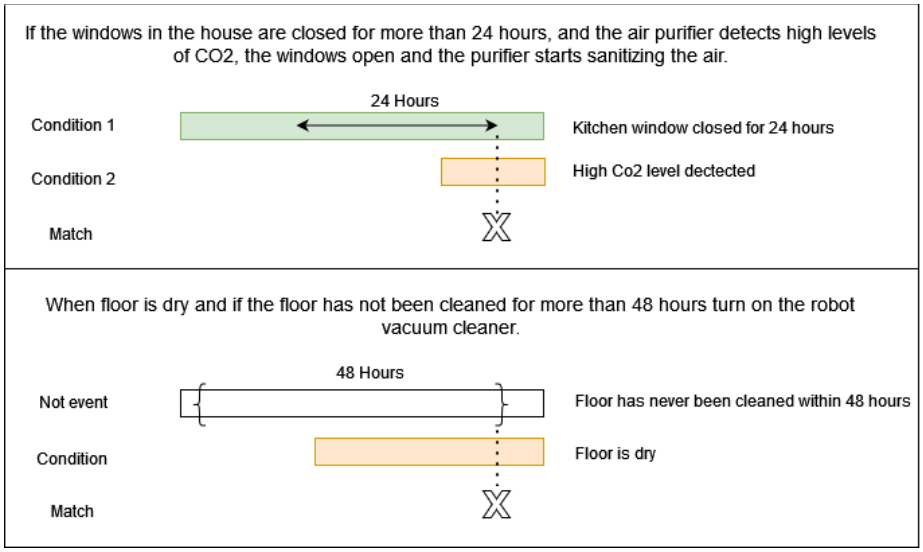
Standard and non-standard rule constructs. To further analyse the rules constructs, we needed to define some properties to allow us to discriminate between “standard” and “peculiar” user-generated automation. After reviewing the relevant literature to understand how automations are typically composed, we set the list of properties that defines a “standard” rule, which is:

- Rules comprised by one or more events joined by the “or” operator, with optionally one or more conditions joined by the “and” or the “or” operator, and one or more actions, joined implicitly by the “and” operator, or by another operator if specified.
- Alternatively, rules comprised of one or more conditions joined by the “and” or the “or” operator, and one or more actions, joined implicitly by the “and” operator.

- Rules including basic operators to characterize a value, such as equal, more/more equal than, less/less equal than, between (e.g., between 9 AM and 10 AM), and the negation and the "different from" operators (e.g., if it is not raining).
- Rules that do not require more structured (such as making another event check after an action) and advanced constructs (e.g., call to external routines that synchronize actions).

Overall, 50 out of the 204 automations produced were "non-standard". To describe these functionalities, participants used and modified the standard keywords and constructs, for instance adding "and (after 5 minutes)" as "next operator" between two actions or adding a new event after an action. The recurring constructs identified in these peculiar rules will be discussed in the following.

Timing aspects of triggers. The most conspicuous aspect is the frequent presence of automations that require some advanced temporally based check. Overall, 19 of these non-standard rules identified needed an additional precise temporal specification in the trigger part. Among these, common cases were automations requiring a check (that may eventually lead to rule activation) to be performed at the end of a defined period. This interval can be indicated by a condition (Figure 4, example 1) or by the non verification of an event (Figure 4, example 2).



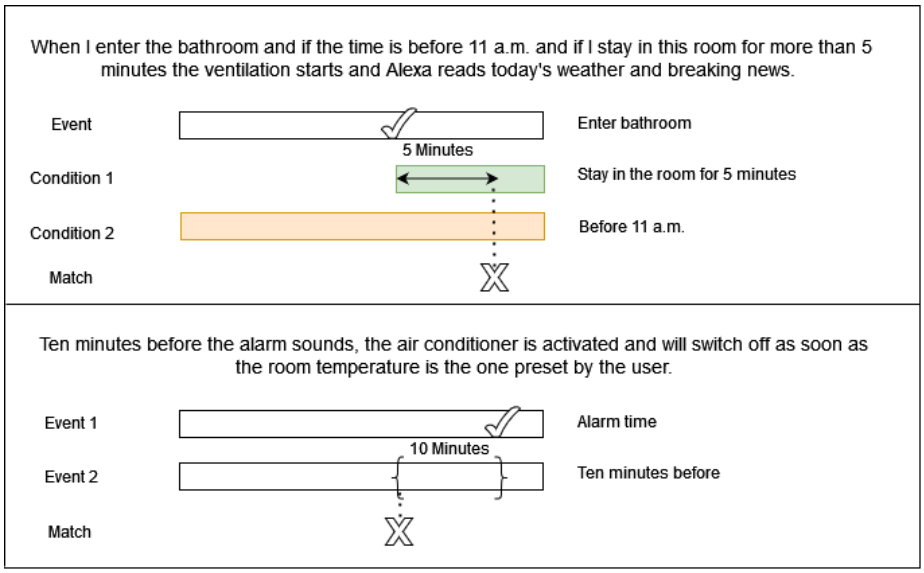


Fig. 4. Scheme of the common timing aspects of triggers identified in the user-created automations.

From a structural perspective, in these cases, the same automation can be written using a condition or negated event (referring to the first two examples in Figure 4, the alternatives are “the windows have never been opened in the last 24 hours” and “The robot cleaner has been off within the last 48 hours”). Also, in some cases, the temporal audit was made in relation to another event. These checks were described both as “forward” (see Figure 4 example 3, where the system should check for the state of a condition within a time frame, starting from when another event occurred) or “backward” (whether the temporally bounded event verifies in an interval before the other one, as the example 4 in Figure 4).

Non-explicit timing aspects of triggers. Another timing aspect of the trigger part was found in automations that did not set a specific time interval but used or implied some timing operator, such as a sequence between the parts of the trigger (for instance indicated by “after that” or “before that”) or a time buffer. Without this specification, the rule could not be activated or would not behave as the user expected. For instance, “I don’t want the stoves to be turned off each time I exit the kitchen, but only when I forget to turn them off.” This automation hence implies a period where stoves are on and unattended. We considered rules where participants used the terms “forgot” or “remains” to describe these types of behaviours (4 instances), and automations that require that the check between the triggers not be immediate but wait for the other trigger to happen, or occur in a non-fixed period before the other (2 occurrences). Another used description is to perform an action at the end of a state (4 occurrences). Below are some examples of user-created automations with these constructs.

- If the refrigerator remains open, send a message to your cell phone.
- When (after that) the car enters the garage, and if the parked car does not move for 10 seconds, the garage door closes automatically, and the lights come on.
- When I finish taking a shower, turn on the stove in the room at 25°.

Timing aspects of actions. Temporal aspects can also be found in actions (8 occurrences). This type of rule involves for instance setting a delayed notification or programming a set of actions in the future (such as opening the blinds gradually at time intervals). For example:

- When it is 21:00, if the user has not yet called the home number of his grandparents, send a text message with the words «How are you?», then after 5 minutes turn off the TV and start a call to their number.
- If the backrest is raised and our user does not get up, then after five minutes the alarm goes off, and the backrest starts to vibrate.

Programming-like Constructs. A different class of structures (6 occurrences) involves some more advanced, programming-like constructs, such as "if-then-else" (when the trigger verifies, if this condition is true then do this, else if this other condition is true…), parenthesis to defining compound Boolean conditions in the trigger part, or counters and accumulators. Below are some examples of these constructs (the first is an "if-then-else" rule, the second needs parentheses, and the last one use a counter of how many times an action has been activated during the day). For example:

- When the user passes his hands under the automatic soap dispenser, if the soap is finished a small red light turns on; if instead the soap is present a small green light turns on and the soap is dispensed.
- When it is 11 pm on a weekday or 2 am on a public holiday, if the user is not at home, turn off the lights, and lock the windows and the door.
- When the button on the bowl is pressed and if daily food dispenses are less than three then the bowl speaker voice notification says "Bravo" and releases food.

Routines and Rules Concatenations. In this group of automations (n = 7) participants envision the possibility of using advanced synchronizations of triggers or actions, such as a digital assistant that can interact with online services to buy the required medicines after the doctor sends the prescription, or create routines consisting of other triggers that have to be checked after the actions of the first part of the automation have completed. For example:

- If the house temperature is not higher than 17°, turn up the heating and then check the air humidity level, if the humidity is lower than 35% activate the humidifier.
- If the user has not taken the medicines between 08:00 and 09:00, send a notification to the smartphone, and if the user has not given any confirmation of reading, activate the voice notification from the Bluetooth speakers.

- When the user is unwell, if the Smart Watch reports a body temperature above 37°, send a text message to the doctor to get a prescription for a medicine and (then) contact the pharmacy to have it sent home.

Groups. In three rules, participants imply the possibility of creating groups for types of objects and services, for instance, “all the children’s devices” or “all the social media platforms”.

- When it's 11 pm, and the kids' devices are in use, turn off the lights and send a reminder to turn off devices and go to bed.

5 Discussion and conclusions

The analysis of the automation rules defined by the participants allowed us to identify some general implications that can support the design of new tailoring environments for TAP that are more capable of matching users' needs. The first is that more attention should be paid to how users can define the timing aspects between rule elements, especially in the trigger part. Participants expressed many behaviours that require different types of temporal occurrences, but often commercial products such as IFTTT do not allow the specification of temporal relations such as “at the end of a period”, or “if this does not occur within this period”. Furthermore, these aspects have also received little attention from research. A first approach for defining complex timing relations that also consider how end users can understand and use these constructs in a smart home context is CCBL [31]. The prototype handles temporal aspects using a subset of Allen’s interval Algebra [1] operators, but can only manage automations based on a hierarchical organization of states. Another possible approach could be using the set of temporal operators for detecting complex events defined in [19] and tested in [2, 21]. In addition to the temporal aspects, participants defined automations that require concepts more advanced than the ones usually definable with ECA syntax, such as the “if-then-else” construct, complex Boolean conditions requiring parenthesis, using iterators or performing some further event check after the activation of an action. Although these possibilities can enrich the expressive capabilities of the TAP rules, it should be considered whether the introduction of these further capabilities would impact the easiness to use of the platform for all users. For instance, a balanced solution could offer both simple and more complex modelling options [7, 32]. This makes also critical the research of tools to simplify the management of automations, such as debuggers [11, 24] or visualizations [36] for making the rule-checking sequence more perceivable and graspable. Another related aspect that emerges is that participants used different approaches for describing the behaviour they wanted to implement using natural language. For instance, they described conditions linked to sustained actions using both the “while” and the “if” terms. Also, no uniqueness emerges for selecting a keyword for events and conditions, and although the terms “when” and “if” were the most frequently used, participants sometimes mixed them. Indeed, it is still not completely clear how to transmit the fundamental concept of the distinction between events and conditions. Recent work [15, 35] started

to clarify which terms lead to a better understanding of this distinction, but also other linguistic cues could be used to guide users in shaping effective mental models. These observations should be taken into account when specifying how the platform displays the automations, or in designing conversational agents for creating automation rules. Regarding the user-specified context, a limited number of participants defined this field. Although not particularly used in this study, this specification could be helpful in realistic situations to understand when a rule should be enabled. For instance, one user indicated “school time” as context for an automation that activates Alexa to ask the student whether he really wants to play with the Playstation when it is study time, which should be deactivated during the summer, hence reducing the possible interferences [18] and simplifying the monitoring and debugging of the active automations. Concerning the rule goals, an observation is the wide-spreading of the “comfort” goal. Another is that only the “well-being” and the “comfort” goals have a complete overlap of the top-3 most frequent functionality classes. This means that the final goal of a rule can give us some hints on the functionalities that will be required. Together with the patterns between the functionality classes that emerged, these are clues that a recommender system exploiting these patterns could be useful to speed up and simplify the rule composition.

Regarding the directions for future work, this analysis shows that there is a gap between the apparent simplicity of the trigger-action rules and the nuanced functionalities that it could be possible to implement with them, for example by using in combination events, conditions, and temporal operators. There is hence the need to further investigate how to enable users to take advantage of these features. Another aspect to investigate is whether a solution where these interactions are explained in detail (for example, showing how the system carries out the "check" of an automation) is more effective or whether it would be better to abstract this complexity from the users and manage it automatically. Also, how to personalize these approaches for a specific user (for instance, providing recommendations/explanations and gracefully increasing the available options) should be inquired. Finally, further studies should consider novel approaches to TAP, for instance, combining EUD paradigms with AR/VR and conversational-based interaction technologies, assessing their strong and weak points, and implementing them into novel EUD platforms.

In conclusion, we have presented the design and the results of a study aimed at exploring the functionalities that users expect from smart homes, and the constructs and operators necessary to specify these behaviours. From the analysis of the created automations, we identify some common functionalities that users expect, the long-term goals for which they created these rules, and the relations between goals and functionalities. Furthermore, the analysis provides indications of the “non-standard” constructs necessary to implement these functionalities. We believe that this information can be useful in designing future EUD systems for smart home environments.

Acknowledgements. This work has been supported by the PRIN 2017 “EMPATHY: Empowering People in Dealing with Internet of Things Ecosystems”, https://www.empathy-project.eu/

References

1. Allen, James F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26, no. 11 (1983): 832-843.
2. Augusto, Juan Carlos, and Chris D. Nugent: The use of temporal reasoning and management of complex events in smart homes. In ECAI, vol. 16, p. 778. 2004.
3. Bak, Nayeon, Byeong-mo Chang, and Kwanghoon Choi: Smart block: A visual programming environment for smartthings. In 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, pp. 32-37. IEEE, 2018.
4. Barricelli, Barbara Rita, and Stefano Valtolina: Designing for end-user development in the internet of things. In End-User Development: 5th International Symposium, IS-EUD 2015, Madrid, Spain, May 26-29, 2015. Proceedings 5, pp. 9-24. Springer International Publishing, 2015.
5. Brackenbury, W., Deora, A., Ritchey, J., Vallee, J., He, W., Wang, G., Littman, M. L., Ur, B.: How Users Interpret Bugs in Trigger-Action Programming. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, New York, NY, USA, Article Paper 552, 12 (2019). https://doi.org/10.1145/3290605.330078
6. Breve, B., Cimino, G., Deufemia, V.: Identifying Security and Privacy Violation Rules in Trigger-Action IoT Platforms with NLP Models, IEEE Internet of Things Journal 10(6), 5607–5622 (2023).
7. Brich, Julia, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub: Exploring end user programming needs in home automation. ACM Transactions on Computer-Human Interaction (TOCHI) 24, no. 2 (2017): 1-35
8. Cabitza, Federico, Daniela Fogli, Rosa Lanzilotti, and Antonio Piccinno: Rule-based tools for the configuration of ambient intelligence systems: a comparative user study. Multimedia Tools and Applications 76 (2017): 5221-5241.
9. Chen, Xuyang, Xiaolu Zhang, Michael Elliot, Xiaoyin Wang, and Feng Wang: Fix the leaking tap: A survey of Trigger-Action Programming (TAP) security issues, detection techniques and solutions. Computers & Security (2022): 102812.
10. Corno, F., De Russis, L., Roffarello, A. M.: A high-level semantic approach to end-user development in the Internet of Things. International Journal of Human-Computer Studies 125, 41-54 (2019).
11. Corno, F., De Russis, L., Roffarello, A. M.: Empowering end users in debugging trigger-action rules. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, pp. 1-13 (2019).
12. Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello: How do end-users program the Internet of Things?. Behaviour & Information Technology 41, no. 9 (2022): 1865-1887.
13. Demeure, Alexandre, Sybille Caffiau, Elena Elias, and Camille Roux: Building and using home automation systems: a field study. In End-User Development: 5th International Symposium, IS-EUD 2015, Madrid, Spain, May 26-29, 2015. Proceedings 5, pp. 125-140. Springer International Publishing, 2015.
14. Desolda, G., Ardito, C., Matera, M.: Empowering end users to customize their smart environments: model, composition paradigms, and domain-specific tools. ACM Transactions on Computer-Human Interaction (TOCHI), 24(2), pp.1-52 (2017).
15. Desolda G., Greco, F., Guarnieri, F., Mariz, N., Zancanaro, M.: SENSATION: An Authoring Tool to Support Event–State Paradigm in End-User Development. In HumanComputer Interaction – INTERACT 2021, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen

Petrie, Antonio Piccinno, Giuseppe Desolda and Kori Inkpen (eds.). Springer International Publishing, Cham, 373–382 (2021). https://doi.org/10.1007/978-3-030-85616-8_22
16. Fogli, Daniela, Matteo Peroni, and Claudia Stefini: Smart home control through unwitting trigger-action programming. In Proc. 22nd Conf. Distrib. Multimedia Syst.(DMS), pp. 194-201. 2016.
17. Fogli, D., Peroni, M., Stefini, C.: ImAtHome: Making trigger-action programming easy and fun. Journal of Visual Languages & Computing 42, 60-75 (2017).
18. Funk, Mathias, Lin-Lin Chen, Shao-Wen Yang, and Yen-Kuang Chen: Addressing the need to capture scenarios, intentions and preferences: Interactive intentional programming in the smart home. International Journal of Design 12, no. 1 (2018): 53-66.
19. Galton, Antony: Eventualities. Foundations of Artificial Intelligence, 2005, 25–58. doi:10.1016/S1574-6526(05)80004-5.
20. GitHub, https://github.com/andrematt/trigger_action_rules, last accessed 2023/04/14.
21. Gómez, Rodolfo, Juan Carlos Augusto, and Antony Galton: Testing an Event Specification Language. In SEKE, pp. 341-345. 2001.
22. Huang, J., Cakmak, M.: Supporting mental model accuracy in trigger-action programming. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15, pp. 215–225. ACM Press, Osaka, Japan (2015). https://doi.org/10.1145/2750858.2805830.
23. Manca, Marco, Fabio Paternò, and Carmen Santoro: Remote monitoring of end-user created automations in field trials. Journal of Ambient Intelligence and Humanized Computing (2021): 1-29.
24. Manca, M., Paternò, F., Santoro, C., Corcella, L.: Supporting end-user debugging of trigger-action rules for IoT applications. International Journal of Human-Computer Studies 123, 56-69 (2019).
25. Mattioli, A., Paternò, F.: A visual environment for end-user creation of IoT customization rules with recommendation support. In: Proceedings of the International Conference on Advanced Visual Interfaces, pp. 1-5 (2020).
26. Mi, Xianghang, Feng Qian, Ying Zhang, and XiaoFeng Wang: An empirical characterization of IFTTT: ecosystem, usage, and performance. In Proceedings of the 2017 Internet Measurement Conference, pp. 398-404. 2017.
27. Prange, Sarah, and Florian Alt: I Wish You Were Smart (er): Investigating Users' Desires and Needs Towards Home Appliances. In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, pp. 1-8. 2020.
28. Salovaara, A., Bellucci, A., Vianello, A., Jacucci, G.: Programmable smart home toolkits should better address households' social needs. In: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, pp. 1-14 (2021).
29. Soares, Danny, João Pedro Dias, André Restivo, and Hugo Sereno Ferreira: Programming iot-spaces: A user-survey on home automation rules. In Computational Science–ICCS 2021: 21st International Conference, Krakow, Poland, June 16–18, 2021, Proceedings, Part IV, pp. 512-525. Cham: Springer International Publishing, 2021.
30. Statista, Number of IoT connected devices worldwide 2019-2021 with forecasts to 2030, https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/, last accessed 2023/04/12.
31. Terrier, Lénaïc, Alexandre Demeure, and Sybille Caffiau: Ccbl: A language for better supporting context centered programming in the smart home. Proceedings of the ACM on Human-Computer Interaction 1, no. EICS (2017): 1-18.

32. Ur, Blase, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman: Practical trigger-action programming in the smart home. In Proceedings of the SIGCHI conference on human factors in computing systems, pp. 803-812. 2014.
33. Ur, Blase, Melwyn Pak Yong Ho, Stephen Brawner, Jiyun Lee, Sarah Mennicken, Noah Picard, Diane Schulze, and Michael L. Littman: Trigger-action programming in the wild: An analysis of 200,000 ifttt recipes. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 3227-3231. 2016.
34. Yu, Haoxiang, Jie Hua, and Christine Julien: Analysis of ifttt recipes to study how humans use internet-of-things (iot) devices. In Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems, pp. 537-541. 2021.
35. Zancanaro, M., Gallitto, G., Dina, Y, Treccani B.: Improving Mental Models in IoT End-User Development. Human-centric Computing and Information Sciences 2, Article number 48 (2022).
36. Zhao, Valerie, Lefan Zhang, Bo Wang, Shan Lu, and Blase Ur: Visualizing differences to improve end-user understanding of trigger-action programs. In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, pp. 1-10. 2020.