Understanding Concepts, Methods and Tools for End-User Control of Automations in Ecosystems of Smart Objects and Services

Margherita Andrao [1][0000-0003-2245-9835], Fabrizio Balducci [2][0000-0003-1174-4323], Bernardo Breve [4] [0000-0002-3898-7512], Federica Cena [3][0000-0003-3481-3360], Giuseppe Desolda[2][0000-0001-9894-2116], Vincenzo Deufemia [4][0000-0002-6711-3590], Cristina Gena[3] [0000-0003-0049-6213], Maristella Matera[5][0000-0003-0552-8624], Andrea Mattioli[6][0000-0001-6766-7916], Fabio Paternò[6] [0000-0001-8355-6909], Carmen Santoro[6][0000-0002-0556-7538], Barbara Treccani[1] [0000-0001-8028-0708], Fabiana Vernero[3][0000-0002-8093-5943], Massimo Zancanaro[1] [0000-0002-1554-5703]

[1] University of Trento, Trento, Italy
{margherita.andrao, barbara.treccani, massimo.zancanaro}@unitn.it
[2] University of Bari, Bari, Italy
{fabrizio.balducci, giuseppe.desolda}@uniba.it
[3] University of Torino, Torino, Italy
{cena, cristina.gena, fabiana.vernero}@di.unito.it
[4] University of Salerno, Salerno, Italy
{bbreve, deufemia}@unisa.it
[5] Politecnico di Milano, Milano Italy
maristella.matera@polimi.it
[6] CNR-ISTI, HIIS Laboratory, Pisa, Italy
{andrea.mattioli, fabio.paterno, carmen.santoro}@isti.cnr.it

Abstract. The continuously increasing number of connected objects and sensors is opening up the possibility of introducing automations in many domains to better support people in their activities. However, such automations to be effective should be under the user control. Unfortunately, people often report difficulties in understanding the surrounding automations and how to modify them. The goal of this paper is to provide a multi-perspective view of what has been done in terms of design, tools, and evaluation in the area of end-user control of automations in ecosystems of smart objects and services. For each aspect we introduce the main challenge, the current possible approaches to address it, and the issues that still need further investigation.

Keywords: End-User Development, Internet of Things, User Experience.

1 Introduction

Over the last few years, we have witnessed a wide diffusion of the so-called smart objects, which have become relatively common in our daily environments. These objects are characterised by the presence of integrated sensors and actuators, and the capability of exchanging data over the Internet. Through the pervasiveness of these technologies, the vision of an Internet of Things (IoT) has become a reality, where physical objects are capable of communicating among themselves and with people and can sense and react to changes in the environment. According to Statista[1], this trend will continue to grow in the next few years. In this evolving landscape, many challenges concerning the interaction of people with these digital ecosystems are emerging. A crucial aspect is how to support people to understand and use these technologies to satisfy their information and automation needs. Indeed, there is considerable academic and commercial interest in providing users with platforms to personalise their environments by IoT without requiring them to write code.

Trigger-Action Programming (TAP) is an End-User Development (EUD) approach aimed at allowing people who may not be expert programmers to specify automations based on the “rule metaphor”. In a TAP rule, the trigger part describes the situation (as recognized by sensors or services) that, when occurring, causes the action part (for instance, a change in the state of a device or the activation of a service) to be actuated. There is growing evidence that TAP basics can easily be grasped by end users. Yet, this approach presents nuances that become apparent - and critical - in complex and realistic situations. For instance, configuring smart environments with multiple active automations [22] or dealing with temporal sequences are complex tasks to master by rules. Even the formulation of a single rule can be challenging, because of the interactions among its different parts. Other problematic aspects can emerge for security issues from automations that do not execute as the user expects. There is therefore the need to better focus on the end-user creation and management of automations in everyday environments. This is a required step to fully leverage the potential and social benefits of the interaction of people with smart environments [63].

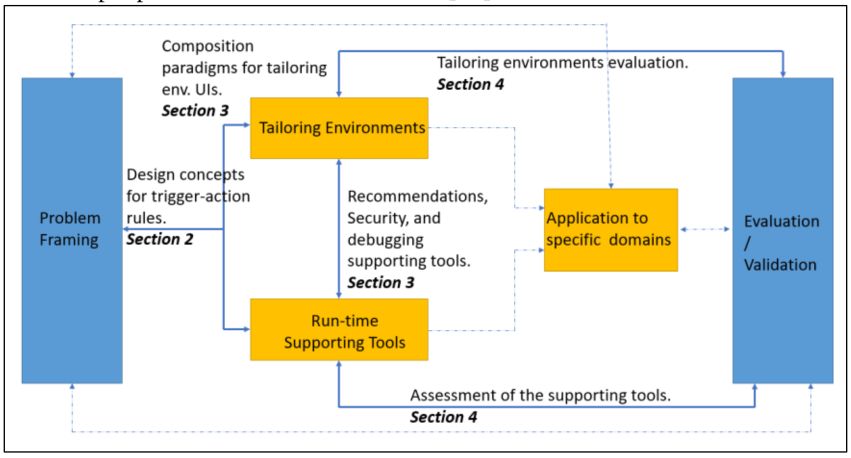
Composition paradigms for tailoring env. UIs.
Section 3
Tailoring environments evaluation.
Section 4
Tailoring Environments
Problem Framing
Design concepts for trigger-action rules.
Section 2
Recommendations, Security, and debugging supporting tools.
Section 3
Application to specific domains
Evaluation / Validation
Run-time Supporting Tools
Assessment of the supporting tools.
Section 4

Fig. 1. The structure of the EMPATHY Project.

[1] https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/, last accessed 2023/04/12.

In this paper, we report the reflections and the experiences on the research activities pursued as part of the EMPATHY project involving six academic groups in developing EUD solutions for IoT in several domains. Figure 1 shows the structure of the EMPATHY project. There are some parts not considered in this paper indicated with dashed lines. The discussion is structured along three axes: what the main aspects of automation rules to consider when designing trigger-action rule languages (Section2), which type of EUD tools for TAP can be relevant (Section 3), and what metrics and evaluation strategies can be considered to assess the outcome of the EUD work (Section 4). The overall goal is to foster a discussion about the state in the area of end-user control of automation in ecosystems of smart objects and services, along with its current challenges, and possible solutions. This discussion can be useful for improving overall understanding of the state of art and identifying areas that require further research efforts.

2 Design of trigger-action languages

2.1 How to Represent Rule Structures

TAP has demonstrated to be relevant in emerging IoT ecosystems as a paradigm that while it does not seem to require specific programming or algorithmic abilities, it allows for the definition of varying behaviours that users can find useful [70]. Triggers can be formulated in terms of events and conditions, and their expression is a key aspect to consider to avoid users creating automations resulting in unexpected behaviours [41]. Several studies investigated how to extend the TAP syntax using more flexible rule formulations. For instance, Desolda and colleagues [30] describe a model with operators for allowing the specification of multiple events and actions, and temporal and spatial constraints on rule elements such as "when" and "where". The model is based on the cause–effect schema, enriched with questions inspired by the 5W model. An iOS application that allows inhabitants to customise the behaviour of a smart home with rules and scenes is introduced by Fogli and colleagues [35]. The goal is to propose an approach capable of modelling automation with enough expressive power through the event-condition-action syntax while unwitting this complexity, by guiding users during the definition of the various rule parts. The work of Salovaara and colleagues [63] aims at discovering “what smart-home automation do technically competent families actually ideate, implement, use, and – most importantly – need?”. To this goal, the study involved various steps (creativity workshops, home automation toolkit installation, and actual use of the environment). Participants could define automations using an extended TAP syntax supporting advanced operators (e.g. time counters, accumulators). Corno and colleagues [21] analysed the levels of abstraction that can be used to describe automations. They argue that current approaches for connecting IoT devices use a one-to-one, low-level mapping, sometimes dependent on the brand of objects involved, while users should instead be better supported by a higher level of abstraction. For this reason, they introduce the EUPont ontology for EUD in the IoT context, which was integrated into a EUD platform and evaluated with a user study, demonstrating the feasibility and understandability of the approach.

Another recent solution [4] based on the use of semantic properties to simplify the creation of ECA rules has been proposed through a design approach that allows domain experts, not necessarily IT experts, to visually specify, for each IoT device, the semantic properties closer to their expertise. An extension of this work reports the results of an analysis of trade-offs in frameworks for the design of smart environments [3]. The main contribution was a framework consisting of a set of trade-offs identified between various dimensions that characterise the quality of software environments for ECA rule design (Creativity, Workload, UX, Engagement, Utility, Completeness and Ease of Use). A recent extension of this research has also explored the use of a system that, through a tangible user interface integrated with pattern recognition and computer vision techniques, assists Cultural Heritage experts in creating smart interactive experiences by properly tailoring the behaviour of the smart objects involved [6].

2.2 Distinguishing between Events and Conditions

The structure of the temporal aspect of the trigger part of an automated procedure is critical for allowing more evolved behaviours, and it is one of the main sources of errors in TAP [8]. To limit possible ambiguities in the formulation and interpretation of TAP rules, it is necessary to clearly support the correct definition of temporal specifications. Brackenbury and colleagues classified the temporal aspects of the triggers, identifying three different compound structures (event-event, event-condition, condition-condition). EUD platforms can implement all these different combinations, however, it is not immediate how to apply them in a way that is clear for users. For instance, the event-event combination makes sense only when the "OR" operator is used (because an AND would imply that the events should occur at the same time, which is unlikely), or when a time window for the second event to occur is specified. The event-condition structure, although non-ambiguous, can be misunderstood by users.

Huang and Cakmak [44] highlighted that it is possible to distinguish at least two types of triggers based on temporal features: instantaneous occurrences (events) and occurrences protracted over time (conditions on states). They noted that users often confused events and conditions on states and remarked that the lack of distinction between different trigger types may create ambiguities (e.g., the interpretation of when exactly triggers will occur) and undesirable outcomes. The considerations raised by Huang and Cakmak have been taken into account in [41], where a rule editor distinguishing between events and condition triggers is described. Events are associated with the ‘becomes’ keyword (linked to when an attribute changes its value), while conditions are related to the ‘is’ keyword. An interactive natural language description of the automation is used to further bring attention to this distinction.

Language use implicitly supports the distinction between these different types of occurrences [13, 59]. De Vega and colleagues [29] observed that, when temporal sentences involve two simultaneous occurrences (e.g., conditions on states and events), one is generally interpreted as the main one, while the other one is considered as the ground (i.e., the context). They also showed that: (i) participants tended to perceive the longer occurrence as the ground; (ii) they found this occurrence more acceptable when introduced by “while” vs. “when”. In some recent studies, temporal conjunctions (i.e., “while” before conditions on states and “when” before events) have been used to distinguish more intuitively conditions on states and events in trigger-action rules. Gallitto et al. [36] and Zancanaro et al. [76] observed an improvement in the accuracy of participants' mental models, and in understanding rule behaviour when different conjunctions ("when" and "while") introduced triggers with different temporal features (events and conditions on states) instead of using more general conjunction ("if"). Indeed, using "when" and "while" to introduce the trigger part of the rule might facilitate users in distinguishing events and conditions on states and understanding their semantics. Desolda et al [31] designed and evaluated an authoring tool for EUD, which adopted "while" and "when" to explicitly support the distinction of events and conditions on states in trigger-action rules creation. Their results revealed that the when-while structure is effective and manageable by naive users without complicating the tasks. Natural language description and the when-while structure were also employed to create EUD systems for therapists [2] and math teachers [1]. According to all these studies, employing the when-while rule structure allows a clear representation of different types of triggers.

Future studies should further investigate other linguistic cues (temporal conjunctions, syntactic order, verbal structure) for designing EUD interfaces to support and guide non-expert users in taking effective mental models.

3 Tools for EUD

3.1 Composition Paradigms

In general, composition paradigms indicate how tools represent the relevant concepts and interact with users, and how they support the development process. Several composition approaches are possible for creating and modifying rules in TAP: form-based approaches (based on Wizard-like support), block-based, dataflow, conversational, and augmented-reality are the most investigated so far. The form-based approach has been used in commercial platforms such as IFTTT, Zapier, Alexa app but also in various research-based approaches [e.g. 30, 41]. The composition of automations using this approach exploits visual structures that conceptually group functionalities and the filling of fields. Such approaches have the advantage to guide the user through a well-defined procedure, and, as such, they are typically regarded as easy to use. However, their rigidity can be perceived as too limited in some cases, and therefore not suitable for dealing with more complex scenarios and task automation [7].

Another approach is block-based programming, which has long been considered in EUD for various domains. A well-known example is Scratch [60], a programming environment primarily targeting children. Using a block-based approach, the definition of a program occurs by connecting blocks of various sizes and shapes, dragging and dropping them in a workspace area. Being less restrictive than form-filling languages, applications based on block programming can leave more space for users’ creativity.

An example in the TAP domain proposed a block-based rule editor [54] where the event-condition distinction is highlighted using a box with a corresponding icon and description. Inspired by the block-based paradigm, and aimed to amplify educational contexts, Gennari et al. [40] introduced IoTgo, a "phygital" toolkit offering smart cards representing sensors and actuators, and hardware and software tools (an ad-hoc scanner and web app), for reading cards and automatically generating scripts running on the most common micro-controllers. The aim of this toolkit is to empower younger generations to explore, empathise with the design context and ideate novel smart things, to deepen IoT programming and communication with certain design patterns.

One further category of visual language commonly adopted in EUD is data-flow. Differently from the previous approaches, the process-oriented nature of data-flow languages makes them more suitable to represent complex use cases [11]. Process-oriented notations can provide more expressiveness, which however is often coupled with complex user interfaces.

Although visual-based paradigms are common interaction modalities and have been extensively studied in the literature, they present some weak points that can make the tools not engaging and immediate to understand. For instance, selecting which items to use in a rule can be time-consuming [41], a source of errors [38], and require users to know some specific technical details.

Recent technological progress provides novel functionalities that can be used to allow for different interactions with devices and services: among others, the conversational paradigm and augmented reality are promising alternatives. An example of a tool that supports the conversational interface is RuleBot [37]. The platform operates by linking the user input to possible triggers and actions, using the defined intents and additional context information. After the first welcome sentence from the chatbot, the user can enter a rule element or an entire rule. The bot provides feedback and asks for further information to complete the automation, if necessary. The system uses natural language techniques to split the user input in parts that are analysed with the support of DialogueFlow for identifying intents and entities. Another approach based on the conversational paradigm is HeyTAP [24], which proposes a set of IF-THEN rules relevant to user's inputs, also considering further, optional higher-level preferences expressed by the user (i.e. energy saving, security). These concepts are modelled using an extension of the EUPont ontology model.

Another technological innovation that can be used to control smart environments is Augmented Reality (AR). The idea of AR is to create an enhanced space where digital components are blended in a natural way in the users' field of view. Visualisations can make perceivable relations between the devices and decrease the cognitive distance between the physical objects and the representation used for rule creation. In the context of tailoring environments, AR can be used to allow for opportunistic and situated rule editing. A solution to support user control of automations using the smartphone as AR device is SAC [5]. It allows end users to frame a relevant sensor or object using the smartphone camera, obtain the rules currently associated with it, edit them or create a new one. It also allows monitoring of the state of sensors or rules involving a whole environment. An approach with a different goal and target users is MagiPlay [67], a serious game aimed at cultivating children’s computational thinking through the personalisation of an augmented environment. AR is used to allow young learners to capture object visualisations and then combine 3D bricks into automation rules. In this area an interesting future work is the design of augmented reality solutions able to show recommendations of possible automations. Another important future direction is to explore how applications of transformers [75] and Large Language Models, such as ChatGPT, will impact the EUD for the IoT ecosystem, for instance, assessing whether such applications are capable of accurately generating automation programs that mirror user intents expressed using natural conversation.

3.2 Security

The definition of automations by users without programming experience can pose serious risks both for the privacy of the user and/or the security of the smart environment [17]. In fact, rules can trigger unexpected behaviours that may be unnoticed by non-technical users [77, 72]. The need for mitigating such risks foster studying and classifying the type of inconsistencies that can be produced [68]. Furthermore, efforts have also been spent on the definition of specific solutions that can support end-users in the identification of security and privacy risks at design time [10, 57]. Chen et al. [15] have proposed a threat model that indicates the various levels at which security issues can emerge in environments where TAP platforms are deployed:

- *Network sniffing*: an attacker can monitor traffic in the IoT network to leak device state or sensitive data from the TAP platform.
- *Privilege escalation attack*: an attacker can exploit over-privileged vulnerabilities to access IoT devices they should not have access to.
- *Malicious automation*: an attacker can create a malicious automation to manipulate triggers and actions to leak sensitive data from IoT device settings.
- *Rule logical attack*: an attacker can manipulate TAP rules to harm the user, cause damage, or facilitate a real-world crime.
- *Unintentional rule logical attack*: legitimate users can unintentionally harm themselves or cause damage by creating TAP rules with logical errors.

Given the high popularity TAP has acquired in the last years, researchers have also focused on investigating how much the users are aware of and concerned about the severity of the risks that might arise from the unintentional creation of risky procedures. The study conducted by Saeidi et al. [62] involved an online survey of 386 participants who were presented with 49 popular rules from the IFTTT platform that had potential security and privacy risks. Participants were asked to evaluate their level of concern for each rule on a scale from 1 to 5, and the results showed that participants were not fully aware of the risks. The top 5 rules for which users expressed the most concern involved acquiring, processing, or sharing their location. The study suggests that there is a need to support users in understanding privacy and security risks while creating trigger-action rules. A recent approach [10] has proposed a classification model for identifying security and privacy violation of rules with one trigger and one action, while a further

contribution [9] presents an explainable AI solution that offers textual explanations for the potential hazards linked to a risky rule. Further research is required to explore other areas, such as providing users with effective ways to mitigate the risks associated with these rules.
Another important aspect analysed in Saeidi [62] is the impact that contextual factors, i.e., the addition of information related to the execution context in which the rules might execute, have on the overall ability of end users to specify the rules. The results demonstrated that, if asked to focus on a specific and detailed context in which a certain rule could be activated, the users may significantly increase their attention to risk factors. Among the contexts which impacted the most, the authors found that security and privacy implications resulted particularly evident, i.e., which entity can activate the rule and who can observe its action. The results of this study suggest investigating mechanisms for providing contextual information of the defined rule. As an example, a TAP might generate contextual factors based on rule information and user's habits.

3.3 Recommendations

Recommender systems (RS) are designed to help users make better choices from large content catalogues [46]. These systems can show users a personalised list of items tailored to their preferences, which can be derived from implicit (such as clicking, purchasing or device usage behaviours) or explicit feedback (a direct rating on an item). Since the creation of an IoT automation is not a "one-shot" operation but consists of multiple selection and configuration steps [26], recommendation support can be used to help users by providing them with relevant objects or services during the various phases. Two main strategies for generating recommendations can be identified in this context. The first is using automations previously created by other users as the building blocks for providing relevant recommendations to the current user. Another viable strategy is to mine some frequent patterns from the user behaviours and derive one or more rules to automate this behaviour. For the former approach, a relevant example is RecRules [23]. The authors put forward a hybrid collaborative and content-based recommender system that considers semantic features to suggest automations based on their functionality (the final purpose for which the person is creating the automation). This solution leverages a reasoner to enrich the automations of semantic information. Through the hierarchical structure of an ontology, a structure of classes and subclasses is applied to items, allowing to uncover functionality and technological relations between them. Then, a collaborative filtering mapping is applied to leverage the community's appreciation or not appreciation of rules. Finally, a learning-to-rank algorithm is trained on the enriched data.

Early examples of the latter approach can be found in PBE (Programming By Example) systems such as EAGER, Dynamic Macro, and APE [27, 53, 61], which continuously observe user's actions to find repetitions over which they can learn a looping program to complete the user's task. Since automation was recommended directly within the user's workflow, users programmed the system without necessarily being aware of it. A more recent example of this strategy is RuleSelector [66], where a mobile interactive tool is used to allow users to select TAP rules from a short summarised list.

The engine of RuleSelector first uses frequent itemset mining to discover contexts that often occur together (such as “watch TV” and “at home”). From these contexts, a list of candidate rules is generated using three selection metrics (confidence, contextual specificity, and interval count). The last step of the algorithm is to summarise the rule list using a total action coverage criteria defined by the authors. Another aspect to consider is whether to generate and show complete rule recommendations, or just rule parts that fit the automation a user is currently editing in a sort of autocomplete fashion. It should be noted that while the step-by-step approach is easy to integrate into the rule composition process, and hence can be beneficial for beginners, the full rule suggestions provide a wider overview of the personalization capabilities [55]. In this area, an important direction for future work is how to combine state-of-the-art recommendation methods with context and user data. The integration between the recommendations generated by automations and those derived from usage patterns can provide more effective results.

Automation rules datasets for recommendations. To generate relevant recommendations and, in general, to provide intelligent support to users, EUD platforms include some “sub-systems” devoted to analysing previous data to extract patterns or other useful information. These sub-systems rely on collected datasets of information relevant to the specific goal of the platform. Regarding recommendations for smart home personalization platforms, the crucial data source is the automation rules dataset. The first publicly available dataset of (IFTTT) automation rules was crawled and curated by Ur and colleagues [69]. The authors scraped all the programs shared publicly on the IFTTT website at that time (67169 rules) and analysed them to investigate the practicality of letting average users customise smart-home devices. The generated recipes dataset contains the following fields: an “if channel” and its trigger condition and any parameters, a “then channel” and its selected action and any parameters, a recipe id, an author id, and statistics on when it was shared and how many other users have activated it. In a subsequent study [70], the authors proposed a new version of the dataset containing the 224,590 programs shared publicly on IFTTT as of September 2015. The dataset includes recipes from more than 100.000 users (most of whom created only one or two of them) and contains many duplicate programs. The dataset[2] was publicly released. Mi and colleagues [56] collected data from the IFTTT website for six months (from November 2016 to April 2017) and performed controlled experiments on these data using a custom testbed. The dataset was generated first obtaining the full list of available services. Then the authors reverse-engineered the URLs of applets’ pages, using this address to fetch more than 300K public applets. For each applet, they obtained its name, description, trigger, trigger service (the service that the trigger belongs to), action name, action service, and add count (the number of this applet being installed by users). The collected data and the testbed[3] are publicly available. A more recent effort was carried out by Yu and colleagues [74]. The focus of the dataset is specifically on rules designating smart space interactions. Another goal is to characterise behaviours that users

[2] https://www.upod.io/datasets.html, last accessed 2023/04/12.
[3] https://www-users.cse.umn.edu/~fengqian/ifttt_measurement/, last accessed 2023/04/12.

encapsulate in automation rules in IFTTT. The authors collected 50,067 publicly shared IFTTT recipes, and then filtered them based on their popularity, relevance to IoT applications, and ease of automated parsing. The final dataset consists of 2,648 IoT-relevant rules. This dataset[4], including the preliminary pre-filtered versions and the scripts used to analyse it, is publicly available. The features of the rules in the dataset are rule name, rule id, services (e.g., 'weather', 'hue'), service names ('Weather Underground', 'Philips Hue'), description, service owner (a Boolean indicating whether the rule is provided by the manufacturer), pro features (a Boolean indicating whether a pro subscription to IFTTT is required), and install count. Another recent effort in regard to rule datasets collection was carried out to obtain automation rules in a less limiting format compared to the traditional IFTTT syntax (allowing i.e. to define automations consisting of multiple conditions and actions, or to use the "NOT" Boolean operator). Another objective was to obtain more contextual information about the automation, such as the role of the rule creator (domain expert, platform expert, or student) and the rule's goal (e.g. comfort, health, security). Rules are described by the fields: author, rule name, goal, natural language description, real name (the name of the device or service), parent (a broader classification of the functionality), operator, value, next operator (what connects a rule element to the next one). The dataset currently contains 638 rules and is available on GitHub[5].

3.4 Debugging and Explainability of Automation Rules

The End-User Development (EUD) paradigm has allowed non-technical users without programming skills to customise the behaviour of their devices and applications [28], empowering users and letting them benefit from the potential of IoT. The relative simplicity and applicability of EUD paradigms to IoT – such as TAP - have attracted great interest [58]. One important aspect to consider is the possible issues derived from interferences between multiple automations. Indeed, despite its ease of use, non-programmers still make numerous mistakes in composing TA rules, like loops, inconsistencies, and redundancies [25]. That is important because poor or conflicting rule settings can lead to unsatisfactory or potentially dangerous behaviour for the user. For example, Chen et al. [15] discuss three possible categories of logical errors in these cases, providing associated examples. *Rule prevention* occurs when the execution of one rule unintentionally prevents the trigger of another rule. For example, if rule 1 is "if nobody is home, turn off the smart outlets" with the intention of saving energy, and rule 2 is "if it's 10AM, turn on the smart pet feeder". If the pet feeder is powered from one of the smart outlets and nobody is home at 10AM, the pet will not be fed. *Rule collision* occurs when the actions of two rules are conflicting, for example: if rule 1 is "if the kitchen sensor detects smoke, open the window" and rule 2 is "if it is dark outside, close the window", when there is something burning in the oven (thereby there is smoke in the kitchen) and it is night, the window may not open because of the rule collision. *Unex-*

[4] https://doi.org/10.5281/zenodo.5572861, last accessed 2023/04/12.
[5] https://github.com/andrematt/trigger_action_rules, last accessed 2023/04/12.

pected rule chain occurs when one rule may trigger another rule unexpectedly. For instance, if rule 1 is "if the temperature in my room is below 20°C, turn on the heater in my room", and rule 2 is "if the temperature in my room is above 25°C, open the window". After executing the first rule, the temperature might rise above 25°C and activate rule 2, which can cause the window to open unexpectedly. In the section about debugging and explainable automations we discussed possible ways to address such issues.

Only recently, a few studies have been carried out that focused on the problem of rule errors in EUD and explored debugging approaches to support end-users in customising their IoT devices [42]. However, while many efforts have been directed toward debugging for mashup programming, spreadsheet, and rule analysis, little work has investigated debugging in TAP [22, 28]. Some of these works, inspired by and extending the Interrogative Debugging paradigm of Ko and Myers [47], proposed tools and approaches that allow end-users to simulate their own rules and identify errors [22, 28, 51]. Specifically, these pioneering works developed and tested different EUD interfaces able to simulate the rules created, detect potential errors, and return explanations of those errors to the user to support them in correcting them. Although preliminary, these studies' results seem to suggest that this type of solutions can support end-users in dealing with and better understanding errors in the composition of trigger-action rules. In particular, ITAD [51] provides the possibility to indicate a specific context of use and automatically check whether certain rules can be triggered in the given context, with the possibility to know why or why not that automation can be executed. FortClash [18] has a different approach, using visual timelines associated with contextual elements and rules highlighting when the contextual items change state in order to better analyse them and control possible conflicts.

However, several aspects remain to be clarified. In particular, we still do not know much about how end-users approach debugging and what strategies they adopt [22]. The meagre extant literature on end-user debugging of TA rules is limited mainly to tools to support bug identification, with limited end user involvement and not delving into user and interaction characteristics. Therefore, a possible research direction could be the study of the strategies exploited by users and the exploration of their mental models during debugging tasks. Knowing more about end users' debugging strategies is essential to inform the design of better tools to support this important task [42]. In a recent pilot study [52], the authors explored the strategies adopted by non-programmer users for testing and debugging a trigger-action rule set focusing on the mental models that users initially have in facing a debugging task. Results highlighted different debugging strategies, partly similar to those already observed in novice programmers [33, 34, 48]. Another aspect is the similarity of these issues with those emerging in explainable artificial intelligence: the set of automations created can be considered a kind of black box that should be made transparent to the users who can ask several questions [49] to better understand the resulting behaviour.

4 Evaluation

4.1 Metrics

It seems fruitful to consider which metrics can be used and when some methods are more suited in the context of EUD for IoT. As introduced in section 2.1, one aspect to assess is how the temporal elements of an automation are perceived by users, and whether their expectations and descriptions correspond to the actual automation execution. The counting of errors during the editing of automations can be used to assess different aspects of a tailoring platform, such as whether its design correctly conveys the concepts of events and conditions or if a debugging tool can decrease the issues in pre-defined automations. For this goal, first, the types of errors should be defined, such as wrong triggers or actions, wrong rule parameters, or incorrect specification of logical operators. A severity index for the errors can also be defined. From these, an "error table" can be designed and used by researchers to assign a score to an automation after they agree on its classification. For instance, in [38] four categories of errors were identified within the rules produced by the participants of a user test and thereby analysed. These categories are: the incorrect definition of triggers and of actions; the incorrect association of a trigger with an event vs a condition; the incorrect application of the NOT operator; and the incorrect use of composition operators (AND/OR) to combine different triggers. The authors assigned a weight (1 point indicating a severe error and 0.5 a moderate one) to the errors according to the defined scheme. The severity of an error was established based on the "distance" between the content and the outcome of the automations (generated using tailoring environments that implement different composition paradigms) and their natural language descriptions. For instance, in the rule "When the user is in bed and the bedroom light level is daylight, send a notification to the user and turn off all the lights in the bedroom", using "bed movement" as a trigger instead of "bed occupancy" is a moderate error, whilst not using any bed-related is a severe error. An analysis of the errors in the generation of automation rules has been performed [30], where the mistakes were categorised as 1) wrong events or actions in a rule; 2) wrong parameters in the specification of events and actions; 3) using a wrong logical operator (AND instead of OR). A score between 1 and 3 was assigned to each type of error, where 3 is the most serious one.

Another measurable aspect to consider is time-to-tasks. In this context, time recordings can be helpful to assess whether a tool more efficiently supports end users in carrying out rule creation and modification tasks [12]. For conversational-based approaches, an evaluation can also be based on the number of conversational turns [45]. Another use of time in assessments can be to set a time limit and check within this interval for the number (and eventually variety) of generated automation to check if one approach is able to stimulate creativity more than another. A similar assessment (but without the time limit) was performed in [25]. A final aspect to consider is users' motivation. There are studies (for instance, [39]) that analyse the components that have a positive or negative impact on the intention to use IoT technologies making reference to the technology acceptance model (TAM). Another aspect is to analyse how these technologies are used after their deployment (as in [50]), supporting analytics to show which object or service is found more useful, or examining the actual usage of automation rules, or whether there is a loss of interest in the platform after some time.

4.2 Assessment of the different composition paradigms

Another aspect to investigate is whether a different composition paradigm has an impact on the user experience on a tailoring platform. For instance, Valtolina and colleagues [71] reported on a study evaluating the benefits of a chatbot in comparison to traditional GUI, specifically for users with a poor aptitude for using technologies. They considered applications in the healthcare and smart home fields and found that for the user experience the chatbot application appears to be better than the GUI-based one. A similar comparison has been carried over in the assessment of the RuleBot [37] platform. RuleBot is a conversational agent that uses machine learning and natural language processing techniques to allow end users to create automations according to a flexible implementation of the trigger-action paradigm, and thereby customize the behaviour of devices and sensors using natural language. The usability of the solution was assessed with a user test that presented users with scenarios of increasing difficulties, requiring them to compose from simple to complex rules using RuleBot (chatbot-based) and the TAREME (wizard-based) platform. After the composition task, users had to compile a questionnaire, assigning scores on a 1 to 5 scale to statements associated with each task. These questions were repeated for each tool. It turned out that for simple rules the wizard style was found more efficient than the chatbot because the users were driven to select the relevant elements, while in the chatbot there was some initial conversational turn to allow users to better understand how to formulate the desired rule. Vice versa, in more structured rules, the chatbot was more efficient since users understood how to indicate them and it was thus quite immediate while with the wizard they still had to navigate across the various sections to find the relevant items.

A similar study but related to a different interaction paradigm was performed in [64]. In the study, a platform (HoloFlows) aimed at enabling users to exploit Augmented/Mixed Reality to simplify the modelling and the configuration of IoT workflows is introduced. The platform exploits concepts from the BPM (Business Process Modelling) domain to allow users to automate tasks involving one or more IoT devices. The assessment of the approach consisted of a user study comparing HoloFlows with other two approaches to model IoT processes based on GUI, Node-RED and Camunda Modeler. The dependent variable was the task completion time. The workload of each tool was examined using the NASA-TLX questionnaire. However, it should be noted that it is not always appropriate to compare the performance of users when using a traditional Web-based tailoring environment and when using an augmented-reality approach because there are fundamental differences between them [5]. The Web-based solutions require explicit access to the tailoring platform, whilst a mobile AR platform can be used more opportunistically. Also, the AR app can remove the necessity to browse the logical organization of triggers and actions available in the editor, hence making less relevant a comparison on task completion times. Other aspects to consider when assessing the user experience of an AR application for personalization rules are whether the selected device is appropriate (e.g., smartphone, tablet, or a dedicated device); if the representations used are understandable (e.g., 3D visualizations that reproduce real objects, or more abstract representations); whether the designed way to interact with the objects is easy to understand; what is the added value of the AR approach, i.e, if it enables functionalities that would not be possible or would be challenging with a different user interface (in this regard, Clark and colleagues [16] assessed the interaction with IoT devices in unfamiliar spaces).

4.3 User experience

Another aspect to consider is the impact on the user experience of supporting tools such as debuggers or recommendations. When an IoT platform is deployed in a real environment, the interactions between rules may generate some unwanted situations, such asthe conflictual triggering of actions, or the specified rule behaviour may result different from the intended one [51].. Hence, a debugging support tool that can test the correctness of rules and possibly identify errors in them (e.g. triggers or actions that they might have forgotten or inappropriately added in the current rule specification). A user-oriented evaluation of such tools could be focused on assessing to what extent the debugging can support non-programmer users. For instance, researchers can generate a dataset of automation that presents different types of errors, such as flipped triggers, or actions that cause loops. Then, users are asked to perform some tasks aimed at checking to what extent they can solve issues in bugged rules under two different conditions: by using the tool and by not using the tool ("control" case). A similar assessment for a debugging tool has been carried out [51] in which participants were provided with some tasks (describing the desired outcome situation) and a set of rules to use to reach the behaviour described in the tasks, which presented some inconsistencies (compared to the intended behaviour). In the 'control' case, they had to write down in natural language the result of their analysis, identifying the rule(s) to edit and the kind of changes to do on one or more rules for specifying the expected behaviour. In the other case, they had to edit/fix the concerned rule(s) by exploiting the simulator functionality and the why/why-not buttons developed in the tool. Another aspect to consider is whether using the support tool can lead to a better understanding of the future system's functioning. For instance, FORTNIoT [19] aims to extend the intelligibility of a system also to the predicted future behaviour of an environment, to allow users to better understand what actions will happen and why. The assessment of the platform has been done through remote interviews, where participants had to use the platform with and without the prediction engine. Participants were asked to think aloud and report what will happen in a scenario and why, also using a 5-point Likert scale to express their confidence in the answer.

A user-centered assessment can be performed on a recommendation system specific to this setting. Besides assessing their user experience on the platforms, other aspects that can be evaluated are the perceived relevance, novelty and diversity of recommendations, and overall satisfaction. To provide a reliable measure of the quality of an algorithm, these user-centered metrics should be combined into a single score [32]. A specific way to assess recommender systems can be to ask users to perform some tasks and check during these tasks whether the presented recommendations are selected [43]. Another relevant aspect that can be tested is if the proposed recommendations speed up the rule generation process or make it easier. It should be noted that a recommender system may increase the time to complete a task, suggesting users some interesting alternatives. Hence, it should be analysed how to correctly integrate the recommender system with the users' operation on the tailoring platform to propose a convenient solution. Another consideration is whether recommendations fit the users' desires concerning long-term preferences, such as convenience, sustainability, security, or privacy. This aspect can be assessed with a match with an inserted or derived user profile, using as a baseline an algorithm that does not consider the user profile (as in [73]).

Another not yet widely considered aspect is the impact of psychological constructs (such as personality traits, locus of control, mindset,...) on user perceived recommendation usefulness as well as on their performance in configuration tasks where they could use recommendations. Cena et al. [14] for example carried out an experiment in the context of home automation where they found that the personality traits of Need for Cognition and Self-efficacy may play an important role in assessing the perceived usefulness of recommendations, and they have some relation in conjunction with the performance of the task.

5 Conclusions

This paper discusses recent proposals in the EUD field addressing the control of IoT automations in ecosystems of smart objects and services. The discussion is structured along the main aspects of automation rules to consider in the design phase, the current relevant tools for EUD in smart ecosystems, and which approaches can be used to assess the outcome of the user interaction with these platforms.

The first part of the discussion focuses on how to represent the rule structure, and how to allow users to correctly interpret the timing aspects of the automation triggers. Regarding the former aspect, novel contributions indicate the feasibility of designing extensions and using different TAP abstraction levels while retaining their understandability. There is a gap between commercial applications, usually not offering much flexibility in rule creation, and advanced home automation systems that require programming skills, e.g. Node-RED. This divide should be investigated more, and to this end, it is of crucial importance to better understand what kind of automations users need, and how these personalization intents are expressed, as well as better grasp the users' mental models about TAP [65]. Another characteristic that emerges [20, 35, 63] is that not much consideration has been given to how different people sharing the same space interact with the automations. This can raise conflicts based on different personal preferences, but it also presents new possibilities, for instance socially-oriented automations aimed towards the well-being of families, which require the cooperation of more people to be achieved. This aspect should be considered while designing possible extensions for TAP rules, providing operators, structures and functionalities that also consider these possible situations, while also keeping in mind the heterogeneity of the actors that can co-participate in the creation of the automation programs. Concerning the distinction between states and events in TAP rules composition, recent studies ([36, 75]) take into account that this distinction can be expressed in natural language [13, 59] and observe that the choice of more specific temporal conjunctions (when-while) leads to an improvement in their understanding. Implementations of EUD systems for different use cases [1, 2, 31] employing the when-while rule structure demonstrated that it enabled clear representation of event and condition triggers. Future work should investigate which other linguistic cues EUD interfaces should use to better support users.

We have also investigated the tools for EUD. The first considered aspect is which composition paradigms are currently used for creating and modifying TAP rules. Visual-based paradigms have been studied in the literature, where their main strengths and weaknesses have been analysed. Recent technological progress allows for new approaches exploiting different interaction modalities, such as tangible, conversational and AR. However, there are not many studies where these approaches are compared with visual tools, or where they are analysed in a context of realistic use over longer periods of time. Studies are needed to better assess how they impact the understanding, definition and control of automations in IoT settings and the long-term engagement of potential users with the system. Concerning the security aspects, different levels of threats can emerge from automations rules and can put at risk the privacy of the user and/or the security of the smart environment. Furthermore, these unexpected behaviours may go unnoticed by non-technical users. Some work has started to investigate users' concerns in IFTTT rules [62], and how to classify the different threats [10] also using textual descriptions to explain why a rule can be dangerous [9]. There are however further aspects that impact how users perceive the risks related to rules, in particular the context where rules might be activated. Context-awareness could be helpful in providing personalized recommendations. Current approaches mainly use two recommendation strategies: they rely on automations created by other users or try to mine some frequent patterns from the user behaviours and derive rules to automate this behaviour. A challenging future direction is an approach capable of combining the two strategies, e.g. considering automations created by others but completing them with the values from the context of the user. Concerning the debugging and explainability of TAP rules, some pioneering works started to investigate how to simulate the rules execution, identify potential errors, and return explanations of those errors. Although preliminary, the results seem to indicate that these tools can support users in composing TAP rules and in understanding the errors that may emerge. However, several aspects remain to be clarified, for instance about how end-users approach debugging and what strategies they adopt [22].

The last section considers the approaches for usability and user experience evaluation. It shows that while some general purpose evaluation methods can be applied in this area, there is a need for specific methods that capture its specific aspects, such as how to identify and assess errors in creating automation rules. In general, this area would benefit from more studies on how to motivate non-technical users to create their automations, and in more extended trials in the wild to better investigate the adoption of EUD approaches.

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/. Balducci acknowledges the support by the REFIN grant, POR Puglia FESR FSE 2014-2020 “Gestione di oggetti intelligenti per migliorare le esperienze di visita di siti di interesse culturale”.

References

1. Andrao, M., Desolda, G., Greco, F., Manfredi, R., Treccani, B., Zancanaro, M. End-User Programming and Math Teachers: an Initial Study. In Proceedings of the 2022 International Conference on Advanced Visual Interfaces, 1–3, (2022).
2. Andrao, M., Treccani, B., Zancanaro, M. Therapists as designers: an initial investigation of end-user programming of a tangible tool for therapeutic interventions. In Proceedings of the 2nd International Workshop on Empowering People in Dealing with Internet of Things Ecosystems co-located with with INTERACT 2021, Bari, Italy, Online / Bari, Italy, September 30, 2021(CEUR Workshop Proceedings, Vol. 3053). CEUR-WS.org, 38–42 (2021). http://ceur-ws.org/Vol-3053/paper_8.pdf
3. Ardito, Carmelo, Giuseppe Desolda, Rosa Lanzilotti, Alessio Malizia, and Maristella Matera. "Analysing trade-offs in frameworks for the design of smart environments." Behaviour & Information Technology 39, no. 1 (2020): 47-71.
4. Ardito, Carmelo, Giuseppe Desolda, Rosa Lanzilotti, Alessio Malizia, Maristella Matera, Paolo Buono, and Antonio Piccinno. "User-defined semantics for the design of IoT systems enabling smart interactive experiences." Personal and Ubiquitous Computing 24 (2020): 781-796.
5. Ariano, Raffaele, Marco Manca, Fabio Paternò, and Carmen Santoro. "Smartphone-based augmented reality for end-user creation of home automations." Behaviour & Information Technology 42, no. 1 (2023): 124-140.
6. Balducci, Fabrizio, Paolo Buono, Giuseppe Desolda, Donato Impedovo, and Antonio Piccinno. "Improving smart interactive experiences in cultural heritage through pattern recognition techniques." Pattern Recognition Letters 131 (2020): 142-149.
7. Bellucci, Andrea, Andrea Vianello, Yves Florack, Luana Micallef, and Giulio Jacucci. "Augmenting objects at home through programmable sensor tokens: A design journey." International Journal of Human-Computer Studies 122 (2019): 211-231.
8. 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
9. Breve, B., Cimino, G., Deufemia, V.: Towards Explainable Security for ECA Rules. In: Proceedings of the 3rd International Workshop on Empowering End-Users in Dealing with Internet of Things Ecosystems (EMPATHY), CEUR-WS Vol.3172, 2022, pp.26-30.
10. 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).
11. 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.

12. 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.
13. Casati, F., Castano, S., Fugini, M., Mirbel, I., Pernici, B. Using patterns to design rules in workflows. IEEE Transactions on Software Engineering 26(8), 760-785 (2000).
14. Cena, F., Gena, C., Mattutino, C., Mioli, M., Treccani, B., Vernero, F., Zancanaro, M: Incorporating Personality Traits in User Modeling for EUD. In: 3rd International Workshop on Empowering People in Dealing with Internet of Things Ecosystems, CEUR Workshop Proceedings, 3172, pp. 41-48, 2022.
15. Chen, Xuyang, et al. "Fix the leaking tap: A survey of Trigger-Action Programming (TAP) security issues, detection techniques and solutions." Computers & Security (2022): 102812.
16. Clark, Meghan, Mark W. Newman, and Prabal Dutta. "ARticulate: One-Shot Interactions with Intelligent Assistants in Unfamiliar Smart Spaces Using Augmented Reality." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 6, no. 1 (2022): 1-24.
17. Cobb, C., Surbatovich, M., Kawakami, A., Sharif, M., Bauer, L., Das, A., Jia, L.: How Risky Are Real Users' IFTTT Applets?. In: Proceedings of the Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020), pp. 505–529, 2020.
18. Coppers, Sven, Davy Vanacken, and Kris Luyten. "FortClash: Predicting and Mediating Unintended Behavior in Home Automation." Proceedings of the ACM on Human-Computer Interaction 6, no. EICS (2022): 1-20.
19. Coppers, Sven, Davy Vanacken, and Kris Luyten. "Fortniot: Intelligible predictions to improve user understanding of smart home behavior." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4, no. 4 (2020): 1-24.
20. Corcella, Luca, Marco Manca, and Fabio Paternò. "Personalizing a student home behaviour." In End-User Development: 6th International Symposium, IS-EUD 2017, Eindhoven, The Netherlands, June 13-15, 2017, Proceedings 6, pp. 18-33. Springer International Publishing, 2017.
21. 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).
22. 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).
23. Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello. "RecRules: recommending IF-THEN rules for end-user development." ACM Transactions on Intelligent Systems and Technology (TIST) 10.5 (2019): 1-27.
24. Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello. "HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation." In Proceedings of the International Conference on Advanced Visual Interfaces, pp. 1-9. 2020.
25. Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello. "TAPrec: supporting the composition of trigger-action rules through dynamic recommendations." In Proceedings of the 25th International Conference on Intelligent User Interfaces, pp. 579-588. 2020.
26. Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello. "Devices, information, and people: abstracting the internet of things for end-user personalization." In End-User Development: 8th International Symposium, IS-EUD 2021, Virtual Event, July 6–8, 2021, Proceedings, pp. 71-86. Cham: Springer International Publishing, 2021.
27. Cypher, Allen. "Eager: Programming repetitive tasks by example." In Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 33-39. 1991.

28. De Russis, L., Roffarello, A. M. A debugging approach for trigger-action programming. Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (2018). http:// dx.doi.org/10.1145/3170427.3188641
29. de Vega, M., Rinck, M., Diaz, J. M., León, I. Figure and Ground in Temporal Sentences: The Role of the Adverbs When and While. Discourse Processing 43 (1), 1-23 (2007). https://doi.org/10.1080/01638530709336891
30. 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).
31. 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
32. Epifania, Francesco, and Paolo Cremonesi. "User-centered evaluation of recommender systems with comparison between short and long profile." In 2012 Sixth International Conference on Complex, Intelligent, and Software Intensive Systems, pp. 204-211. IEEE, 2012.
33. Fitzgerald, S., Lewandowski, G., McCauley, R., Murphy, L., Simon, B., Thomas, L., Zander, C. Debugging: finding, fixing and flailing, a multi-institutional study of novice debuggers. Computer Science Education, 18(2), pp.93-116 (2008).
34. Fitzgerald, S., McCauley, R., Hanks, B., Murphy, L., Simon, B., Zander, C. Debugging From the Student Perspective, in IEEE Transactions on Education, vol. 53, no. 3, pp. 390-396 (2010). doi: 10.1109/TE.2009.2025266.
35. Fogli, D., Peroni, M., Stefini, C. ImAtHome: Making trigger-action programming easy and fun. Journal of Visual Languages & Computing 42, 60-75 (2017).
36. Gallitto, G., Treccani, B., Zancanaro, M. If when is better than if (and while might help): on the importance of influencing mental models in EUD (a pilot study). in Proceedings of the 1st International Workshop on Empowering People in Dealing with Internet of Things Ecosystems co-located with International Conference on Advanced Visual Interfaces (AVI), Ischia Island, Italy, 2020, pp. 7-11 (2020).
37. Gallo, Simone, and Fabio Paterno. "A Conversational Agent for Creating Flexible Daily Automation." In Proceedings of the 2022 International Conference on Advanced Visual Interfaces, pp. 1-8. 2022.
38. Gallo, Simone, Marco Manca, Andrea Mattioli, Fabio Paternò, and Carmen Santoro. "Comparative analysis of composition paradigms for personalization rules in iot settings." In End-User Development: 8th International Symposium, IS-EUD 2021, Virtual Event, July 6–8, 2021, Proceedings, pp. 53-70. Cham: Springer International Publishing, 2021.
39. Gao, Lingling, and Xuesong Bai. "A unified perspective on the factors influencing consumer acceptance of internet of things technology." Asia Pacific Journal of Marketing and Logistics (2014).
40. Gennari, R., Matera, M., Morra, D., Melonio, A., Rizvi, M.: Design for social digital well-being with young generations: Engage them and make them reflect. International Journal of Human-Computer Studies, Volume 173, 2023. https://doi.org/10.1016/j.ijhcs.2023.103006.
41. Ghiani, G., Manca, M., Paternò, F., Santoro, C. Personalization of context-dependent applications through trigger-action rules. ACM Transactions on Computer-Human Interaction (TOCHI) 24, no. 2 1-33 (2017).
42. Grigoreanu, V., Burnett, M., Wiedenbeck, S., Cao, J., Rector, K., Kwan, I. End-user debugging strategies: A sensemaking perspective. ACM Transactions on Computer-Human Interaction (TOCHI), 19(1), 1-28 (2012).

43. Gunawardana, Asela, Guy Shani, and Sivan Yogev. "Evaluating recommender systems." In Recommender systems handbook, pp. 547-601. New York, NY: Springer US, 2012.
44. 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.
45. Jain, Mohit, Pratyush Kumar, Ramachandra Kota, and Shwetak N. Patel. "Evaluating and informing the design of chatbots." In Proceedings of the 2018 designing interactive systems conference, pp. 895-906. 2018.
46. Knijnenburg, Bart P., Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. "Explaining the user experience of recommender systems." User modeling and user-adapted interaction 22 (2012): 441-504.
47. Ko, A., Myers, B. Designing the whyline: A debugging interface for asking questions about program behavior. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 151–158 (2004).
48. Li, C., Chan, E., Denny, P., Luxton-Reilly, A., Tempero, E. Towards a framework for teaching debugging. Proceedings of the Twenty-First Australasian Computing Education Conference on - ACE '19 (2019). DOI:http://dx.doi.org/10.1145/3286960.3286970
49. Liao, Q. V., Gruen, D., Miller, S. Questioning the AI: Informing Design Practices for Explainable AI User Experiences. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, pp. 1-15.(2020).
50. 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.
51. 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).
52. Manfredi, R., Andrao, M., Greco, F., Desolda, G., Treccani, B. Zancanaro, M.: Toward a Better Understanding of End-User Debugging Strategies: A Pilot Study. In Proceedings of the 3rd International Workshop on Empowering People in Dealing with Internet of Things Ecosystems co-located with AVI 2022, Frascati, Rome, Italy, June 06, 2022. (CEUR Workshop Proceedings, Vol. 3172). CEUR-WS.org, 31-35 (2022). https://ceur-ws.org/Vol-3172/short6.pdf
53. Masui, Toshiyuki, and Ken Nakayama. "Repeat and predict—two keys to efficient text editing." In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 118-130. 1994.
54. 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).
55. Mattioli, Andrea, and Fabio Paternò. "Recommendations for creating trigger-action rules in a block-based environment." Behaviour & Information Technology 40, no. 10 (2021): 1024-1034.
56. 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.
57. Paci, F., Bianchin, D., Quintarelli, E., Zannone, N. IFTTT Privacy Checker. In: Proceedings of the International Workshop on Emerging Technologies for Authorization and Authentication (ETAA), pp. 90-107, Springer, 2020.

58. Paternò, F., Santoro, C. End-user development for personalizing applications, things, and robots. International Journal of Human-Computer Studies, 131, 120-130 (2019).
59. Pianesi, F., Varzi, A. C. “Events and event talk: an introduction,” in Speaking of Events. New York, NY: Oxford University Press, p. 3-47 (2000).
60. Resnick, Mitchel, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner et al. "Scratch: programming for all." Communications of the ACM 52, no. 11 (2009): 60-67.
61. Ruvini, Jean-David, and Christophe Dony. "Learning users' habits to automate repetitive tasks." In Your Wish is My Command, pp. 271-XIV. Morgan Kaufmann, 2001.
62. Saeidi, M., Calvert, M., Au, A.W., Sarma, A., Bobba, R.B.: If This Context Then That Concern: Exploring Users’ Concerns with IFTTT Applets. In: Proceedings on Privacy Enhancing Technologies 2022(1), pp. 166-186, 2021.
63. 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).
64. Seiger, Ronny, Romina Kühn, Mandy Korzetz, and Uwe Aßmann. "HoloFlows: modelling of processes for the Internet of Things in mixed reality." Software and Systems Modeling 20, no. 5 (2021): 1465-1489.
65. 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.
66. Srinivasan, Vijay, Christian Koehler, and Hongxia Jin. "RuleSelector: Selecting conditional action rules from user behavior patterns." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2.1 (2018): 1-34.
67. Stefanidi, Evropi, Dimitrios Arampatzis, Asterios Leonidis, Maria Korozi, Margherita Antona, and George Papagiannakis. "MagiPlay: An Augmented Reality Serious Game Allowing Children to Program Intelligent Environments." Transactions on Computational Science XXXVII: Special Issue on Computer Graphics (2020): 144-169.
68. Surbatovich, M., Aljuraidan, J., Bauer, L., Das, A., Jia, L.: Some Recipes Can Do More Than Spoil Your Appetite: Analysing the Security and Privacy Risks of IFTTT Recipes. In: Proceedings of the 26th International Conference on World Wide Web (WWW ‘17), pp. 1501–1510, 2017.
69. 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.
70. 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.
71. Valtolina, Stefano, Barbara Rita Barricelli, and Serena Di Gaetano. "Communicability of traditional interfaces VS chatbots in healthcare and smart home domains." Behaviour & Information Technology 39, no. 1 (2020): 108-132.
72. Wang, Q., Datta, P., Yang, W., Liu, S., Bates, A., Gunter, C. A.: Charting the Attack Surface of Trigger-action IoT Platforms. In: Proceedings of the ACM Conference on Computer and Communications Security, pp. 1439-1453, 2019.

73. Yang, Felix, Saikishore Kalloori, Ribin Chalumattu, and Markus Gross. "Personalized information retrieval for touristic attractions in augmented reality." In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, pp. 1613-1616. 2022.
74. 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.
75. Yusuf, I. N. B., Jamal, D. B. A., Jiang, L., & Lo, D. (2022, November). RecipeGen++: an automated trigger action programs generator. In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 1672-1676).
76. 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).
77. Zheng, S., Apthorpe, N., Chetty, M., Feamster, N.: User Perceptions of Smart Home IoT Privacy. In: Proceedings of the ACM on Human-Computer Interaction, Vol. 2, CSCW, pp. 1-20, 2018.