POLITECNICO DI TORINO
Repository ISTITUZIONALE

HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation

Original
HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation / Corno, F., De Russis, L., Monge Roffarello, A.. - STAMPA. - (2020), pp. 1-9. (International Conference on Advanced Visual Interfaces (AVI '20) Island of Ischia (IT) September 28 - October 2, 2020) [10.1145/3399715.3399905].

Availability:
This version is available at: 11583/2829354 since: 2020-10-04T15:45:58Z

Publisher:
ACM

Published
DOI:10.1145/3399715.3399905

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
ACM postprint/Author's Accepted Manuscript

(Article begins on next page)

19 November 2026

HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation

Fulvio Corno
Politecnico di Torino
Torino, Italy
fulvio.corno@polito.it

Luigi De Russis
Politecnico di Torino
Torino, Italy
luigi.derussis@polito.it

Alberto Monge Roffarello
Politecnico di Torino
Torino, Italy
alberto.monge@polito.it

ABSTRACT

In the Internet of Things era, users are willing to personalize the joint behavior of their connected entities, i.e., smart devices and online service, by means of IF-THEN rules. Unfortunately, how to make such a personalization effective and appreciated is still largely unknown. On the one hand, contemporary platforms to compose IF-THEN rules adopt representation models that strongly depend on the exploited technologies, thus making end-user personalization a complex task. On the other hand, the usage of technology-independent rules envisioned by recent studies opens up new questions, and the identification of available connected entities able to execute abstract users' needs become crucial. To this end, we present *HeyTAP*, a conversational and semantic-powered trigger-action programming platform able to map abstract users' needs to executable IF-THEN rules. By interacting with a conversational agent, the user communicates her personalization intentions and preferences. User's inputs, along with contextual and semantic information related to the available connected entities, are then used to recommend a set of IF-THEN rules that satisfies the user's needs. An exploratory study on 8 end users preliminary confirms the effectiveness and the appreciation of the approach, and shows that *HeyTAP* can successfully guide users from their needs to specific rules.

CCS CONCEPTS

• **Human-centered computing** → **Natural language interfaces**; Ubiquitous and mobile devices; User studies; • **Computing methodologies** → **Natural language processing**; *Knowledge representation and reasoning*; • **Information systems** → *Recommender systems*.

KEYWORDS

Trigger-Action Programming, Abstraction, Conversational Agent, Recommender System, Semantic Web, Internet of Things

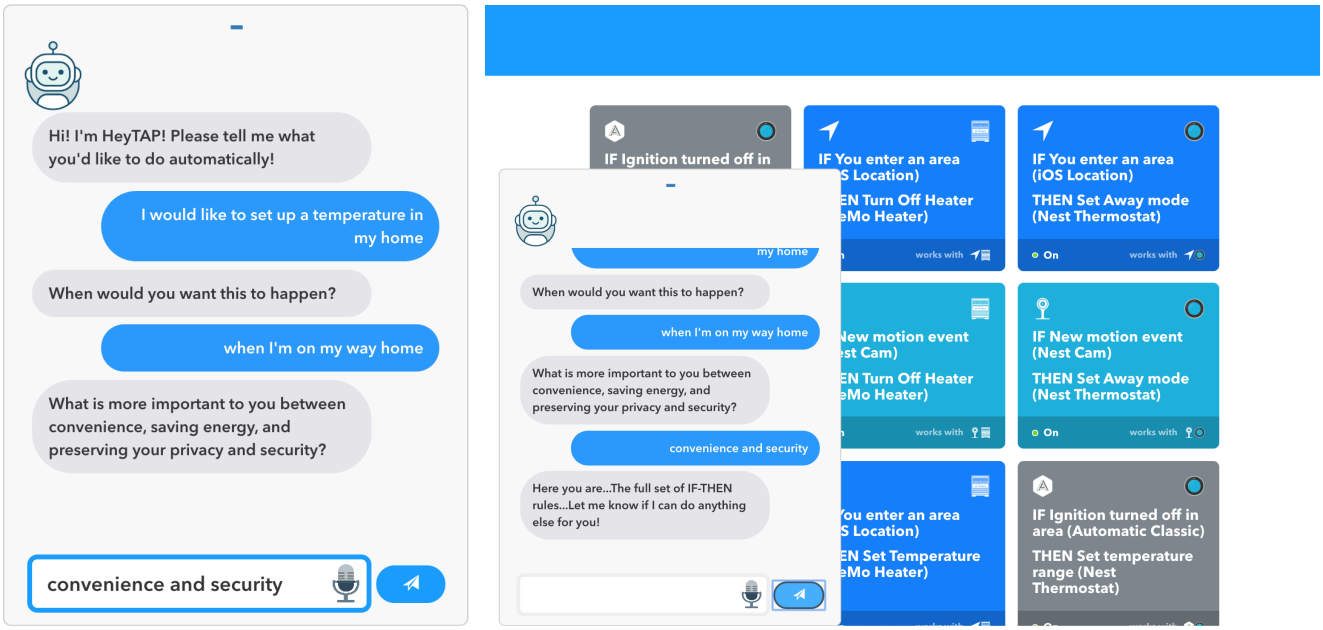
ACM Reference Format:
Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2020. *HeyTAP*: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation. In *International Conference on Advanced Visual Interfaces (AVI '20), September 28-October 2, 2020, Salerno, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
AVI '20, September 28-October 2, 2020, Salerno, Italy
© 2020 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

In the contemporary Internet of Things (IoT) era, people can interact with a multitude of smart devices, always connected to the Internet, in the majority of the today's environments [6]. Smart lamps, thermostats, and many other Internet-enabled appliances are becoming popular in homes and workplaces. Furthermore, by using PCs and smartphones, users can access a variety of online services, ranging from social networks to news and messaging apps. In this complex scenario, the End-User Development (EUD) vision aims at putting personalization mechanisms in the hands of end users, i.e., the subjects who are most familiar with the actual needs to be met [13]. Through visual trigger-action programming platforms such as IFTTT [3] and Zapier [4], users can "program" the joint behaviors of their own *connected entities*, i.e., smart devices and online service, by defining trigger-action (IF-THEN) rules such as *"if I publish a photo on Facebook, then upload it to my Google Drive"*, or *"if the security camera detects a movement, then blink the kitchen lamp."*

Despite apparent simplicity, previous studies [8, 15, 19, 20] highlighted many interoperability, scalability, and understandability challenges suffered by contemporary trigger-action programming platforms. In such environments, smart devices and online services are typically modeled on the basis of the underlying brand or manufacturer [8]: as the number of supported technologies grows, so do the design space, i.e., the combinations between different triggers (*if*s) and actions (*then*s), and users often experience difficulties in discovering rules and related functionality [20]. As a result, trigger-action programming becomes a complex task for people without any previous programming experience [16]. Some previous works, e.g., [8, 13], tackled the identified issues by proposing to move towards a new bread of trigger-action programming platforms supporting a higher level of abstraction, with abstract and technology-independent rules that can be adapted to different contextual situations. With triggers such as "when user is sleeping" and actions such as "illuminate the room", users can personalize their connected entities by saving time and reducing errors, without the need of explicitly programming every single involved technology. While this vision seems promising, however, it is yet unclear how to effectively move from abstract users' needs to the real devices and services needed for implementing them. How can a system decide how to "illuminate" a room? Is turning the lights on the right choice for the user? Does the user prefer to open the blinds, e.g., because she is interested in saving energy?



(a) Intentions and Preferences Input

(b) Recommendations Selection

Figure 1: *HeyTAP* is a conversational and semantic-powered platform for personalizing the behavior of connected entities. First, it allows users to communicate their personalization intentions and preferences (a). Then, it analyzes users' inputs, along with contextual and semantic information related to the available connected entities, to recommend a set of IF-THEN rules able to map the abstract users' needs to real connected entities (b).

In this paper, we present *HeyTAP*, a conversational and semantic-powered platform able to map abstract users' needs to executable IF-THEN rules. By exploiting a multimodal interface, the user can interact with a conversational agent, either by typing or by voice, to communicate her personalization *intentions* for different contexts, e.g., to personalize her room's temperature when she is near home (Figure 1a). By interacting with the agent, the user can also specify her preferences on how to reach the goal of her personalization intention, e.g., convenience and preserving security in Figure 1a. To model such concepts, we extended the EUPont model [7], a semantic representation for End-User Development in the IoT. We exploited the OWL[1] classes and individuals of EUPont to categorize triggers and actions offered by user's connected entities in terms of provided functionality, and to model contextual information, e.g., the devices and services owned by the user and the relative position. Furthermore, we added classes and restrictions to automatically characterize triggers and actions on the basis of the user's preferences, e.g., to discriminate between energy demanding and privacy invasive behaviors. All these semantic information are used to suggest a set of IF-THEN rules that satisfies the user's needs, i.e., intentions and preferences. The user can finally inspect the recommended rules in the multimodal interface and select one or more of them to personalize her connected entities (Figure 1b).

To understand to what extent *HeyTAP* is able to successfully guide participants from abstract needs to actual IF-THEN rules, we ran an exploratory experiment with 8 users. In the study, we challenged participants in freely personalizing a set of connected entities in different contexts. Results confirm the effectiveness of the approach, and show that *HeyTAP* can successfully "translate" abstract users' needs into IF-THEN rules that can be instantiated and executed by contemporary trigger-action programming platforms. Despite participants expressed their personalization intentions with different level of abstractions, in particular, the tool was able to address the 90.63% of the collected needs, by providing IF-THEN recommendations that satisfied the participants. The collected participants' feedback also highlights possible improvements that could inform future works that aim at assisting users in personalizing their smart devices and online services.

2 RELATED WORKS

2.1 Trigger-Action Programming: Opportunities and Issues

One of the most popular paradigm to empower end users in directly programming their connected entities is trigger-action [11, 19]. By defining trigger-action (IF-THEN) rules, users can connect a pair of devices or online services in such a way that, when an event (the *trigger*) is detected on one of them, an *action* is automatically executed on the latter. Trigger-action programming offers a very simple and easy to learn solution for creating end-user applications [5], and trigger-action programming platforms such as IFTTT and Zapier are becoming popular [10, 15].

[1] https://www.w3.org/OWL/, last visited on January 18, 2020

Recently, researchers started to investigate different aspects of these solutions, e.g., through empirical characterization of usage perfomances [18] and large-scale analysis of publicly shared rules [20]. Despite apparent simplicity, indeed, the process of composing IF-THEN rules in trigger-action programming platforms has been found to be a complex task for non programmers [16], and the expressiveness and understandability of solutions like IFTTT have been criticized since they are rather limited [15, 19, 20].

Barricelli and Valtolina [5] analyzed the most popular end-user tools for personalizing connected entities, including IFTTT, and found that some of them "offers a too complex solution for supporting end users in expressing their preferences." By evaluating thousands of trigger-action rules publicly shared on IFTTT, Ur et al. [19] found that the trigger-action approach can be both useful and usable for end-user development in IoT settings like smart homes, but they also found that the level of abstraction end users employ to express triggers needs to be better explored: many users, indeed, express triggers one level of abstraction higher, e.g., "when I am in the room" instead of "when motion is detected by the motion sensor." In another study, Ur et al. [20] found that a large number of users is using IFTTT to create a diverse set of IF-THEN rules, which represents a very broad array of connections for filling gaps in devices and services functionality. According to the authors, however, the continuous growth of supported entities and connections highlights the need to provide users with more support for discovering functionality and managing collections of IF-THEN rules. The analysis emphasizes also the future need of making "IFTTT rules more expressive." Similarly, Huang and Cakmak [15] conducted two user studies to systematically study how different types of triggers and actions, e.g., *states* vs. *events*, influence the understandability of trigger-action artifacts. They found users' inconsistencies in interpreting the behavior of IF-THEN rules and some errors in creating programs with a desired behavior.

2.2 Towards a Higher Level of Abstraction

The aforementioned issues are strictly related to the "low-level" of abstraction of the adopted representations. Contemporary trigger-action programming platforms, indeed, mainly model smart devices and online services on the basis of the underlying brand or manufacturer, thus opening the way to interoperability, scalability, and understandability issues [8]: to program their IoT ecosystems, users need to know all the involved technologies, and they have to define many different rules even if they perform the same logical operations.

To overcome the drawbacks of low-level representations, different previous works [8, 13, 19] envisioned a new bread of trigger-action programming platforms supporting a higher level of abstraction. In the context of smart homes, for example, Funk et al. [12] asserted that we need *"a new approach aimed at first capturing end-users' intentions and potential usage scenarios, then providing this information to a control system that learns to resolve intentions and scenarios for available devices in the context."* Following this need, Ghiani et al. [13] proposed a novel trigger-action programming platform to let end users personalize the contextual behavior of their IoT applications through trigger-action rules. By exploiting an authoring tool, in particular, users can specify trigger-action rules that indicate the desired specific application behavior for the target contexts of use, e.g., *"when user is sleeping, do turn-off bedroom television."* Corno et al. [8], instead, developed EUPont, a high-level representation for IoT personalization that allows users to model abstract trigger-action rules like *"if I enter a closed space, then illuminate it."* Such rules can be adapted to different contextual situations, independently of manufacturers, brands, and other technical details. Besides describing the model, the authors presented its integration in the architecture of a trigger-action programming platform, and they explored the advantages of using the model in the definition of trigger-action rules thanks to a user study. They found that the usage of a higher level of abstraction allows users to define IF-THEN rules with fewer errors and in less time with respect to existing solutions.

While a higher level of abstraction in IF-THEN rules is a promising direction, the identification of the real devices and services to be used to satisfy users' needs becomes crucial. In this paper, we aim at presenting a conversational and semantic-powered platform able to map abstract users' needs to IF-THEN rules that can be executed by available connected entities.

2.3 Programming the IoT via Conversation and Recommendations

By using popular conversational agents such as Amazon Alexa [1] and Google Assistant [2] it is now possible to interact with a variety of different smart devices and online services via conversation. To the best of our knowledge, however, the only example of a conversational system that allows to personalize connected entities through the definition of IF-THEN rules is InstructableCrowd, a research prototype developed by Huang at al. [16]. InstructableCrowd is a crowd-sourcing system that enables users to create IF-THEN rules based on their needs. By exploiting a custom user interface on their smartphones, users can converse with crowd workers to describe some problems they are encountering, e.g., being late for a meeting. Crowd workers can therefore exploit a tailored interface to combine triggers and actions in appropriate IF-THEN rules that are then sent back to the users' phones.

In our work, we focus on a similar goal by trying to automatically map abstract users' needs to actual IF-THEN rules, i.e., without the help of other users such as crowd workers. The idea is to adopt a semantic-based approach to analyze users' inputs and contextual information to *recommend* a set of appropriate IF-THEN rules from which a user can choose. Recommendations, indeed, could be useful to help end users use trigger-action programming systems, and advances in EUD have expanded the opportunities for offering recommendations [14]. In this context, in particular, some recent works investigated how to provide users with recommendations. Yao et al. [21], for example, developed a probabilistic framework to suggest relevant smart "things" to be personalized based on user interests. Corno et al. [9], instead, proposed *RecRules*, a semantic recommendation system that suggests trigger-action rules on the basis of content-based and collaborative information. None of such works, however, explore how to calculate recommendations by extracting users' needs via conversation.

3 *HEYTAP*: ARCHITECTURE AND IMPLEMENTATION

In this section, we first describe the architecture of *HeyTAP*, and we highlight the choices we made in implementing a first prototype of the system.

Figure 2 shows the architecture *HeyTAP*, our conversational and semantic-powered platform for personalizing the behavior of connected entities. The web-based multimodal User Interface (UI) allows the interaction between the user and the conversational agent, and it is responsible of visualizing suggested IF-THEN rules. The UI is implemented through the Angular framework[2], a TypeScript-based open-source web application framework. The conversational agent, instead, exploits DialogFlow[3] as the conversational engine. The *HeyTAP* Server stores all the data related to users, connected entities, and rules, and it interacts with DialogFlow to get users' inputs and provide responses. Furthermore, it is responsible of calculating recommendations on the basis of the collected users' inputs.

HeyTAP support users to move from their abstract needs to IF-THEN rules that involve real smart devices and online services consists in 2 main steps, namely *conversation* and *recommendation*.

3.1 Conversation

By interacting with the conversational agent (Figure 2a), either by typing or by voice, the user first expresses her personalization *intentions*. In this phase, she can use different level of abstraction, and she can refer to different contexts, e.g., she can generically communicate her intention of programming the temperature of a room, or she can refer to a specific lamp in the kitchen to be turned on. Then, the user can communicate her *preferences*, i.e., to specify how to reach the goal of her personalization intention. To model such concepts, we developed EUPont-conversational (Figure 2b), an extension of the EUPont [7] model. We exploited, in particular, the instantiation of EUPont for IFTTT[4]. Such an ontology abstracts details such as brands and manufacturers by categorizing the IFTTT's low-level triggers and actions under a hierarchy of OWL classes that model the provided functionality. Furthermore, the ontology models all the supported connected entities on the basis of their capabilities, and can store contextual information such as entities' position and ownership.

3.1.1 Intentions Elicitation. As exemplified in Figure 1a, personalization intentions are extracted in 2 subsequent phases representing the action and the trigger of an IF-THEN rule, respectively. First, the conversational agent asks the user what she would like to be executed automatically (*action intention*). Then, it asks the user when such an action should be performed (*trigger intention*). If the user specifies a complete personalization intention in the first message, *HeyTAP* uses some predefined keywords, e.g., *"when"* or *"if"*, to split the intention in the corresponding action and trigger. For the sake of simplicity, we focus on detecting simple intentions, i.e., that can be mapped on rules with a single trigger and a single action. This choice is also enforced by the format of the IFTTT rules modeled by EUPont, which do not model trigger conditions and actions involving multiple connected entities. Both action and trigger intentions, in particular, are defined as a set of the following elements, that are automatically extracted by the DialogFlow conversational engine from the user's text:

- a **functionality**, i.e., how users would like to act on their physical and virtual environments. Examples include "increase," "turn off," and "get." To model functionalities, we exploited the OWL classes of the semantic model that classifies all the available IFTTT triggers and actions according to their final goal;
- a **category**, i.e., on which category of connected entities users would like to act. Examples include "temperature," "lighting," and "communication." To model categories, we added new OWL classes to characterize all the available IFTTT triggers and actions;
- an **entity**, i.e., a generic indication of a connected entity type. Examples include "door," "camera," and "social network." To model entities, we exploited the OWL classes that provide a hierarchy of connected entities ranging from physical to virtual objects;
- a **technology**, i.e., a specific indication of a particular technology. Examples include "Philips Hue," "Nest," and "Facebook." To model technologies, we exploited the OWL individuals representing the full set of contemporary IFTTT "services[5]";
- a **where**, i.e., a location in which executing an action or monitoring a trigger. Examples include "kitchen," "home," and "office." To model locations, we specialized the OWL `Location` class into a series of sub-classes modeling homes, rooms, and workplaces;
- a **when**, i.e., a time condition. Examples include "in the evening," "at 10 PM," and "on 27 May." To model time, we used the OWL individuals representing the triggers offered by the Date & Time IFTTT service.

Users can express their intentions with different level of abstractions, by specifying all the described elements, or a subset of them. Table 1 reports some examples of trigger and action intentions. While *"Increase the air quality"* is a very generic action intention that includes a functionality and a category, only, the trigger intention *"when the temperature on my home Nest thermostat drops below 20 degrees"* is way more specific, since it includes a functionality (*increase*), a category (*temperature*), an entity (*the thermostat*), a technology (*Nest*), and a where (*home*). If *HeyTAP* is not able to map an action or a trigger intention onto the modeled elements, it asks the user to reformulate her message. Moreover, the tool explicitly warns the user in case of personalizations that are modeled but not available, e.g., when the available connected entities do not provide a specific functionality.

3.1.2 Preferences Elicitation. Preferences represent a filter on how to implement the user's personalization intentions. We derived the available preferences from the work of Funk et al. [12], that analyzed the temporal, preferential, technical, and social complexity of mapping high-level end-user intents to rules in the smart home

[2]https://angular.io, last visited on January 21, 2020
[3]https://dialogflow.com/, last visited on January 21, 2020
[4]http://elite.polito.it/ontologies/eupont-ifttt.owl, last visited on January 21, 2020
[5]https://ifttt.com/services, last visited on January 21, 2020

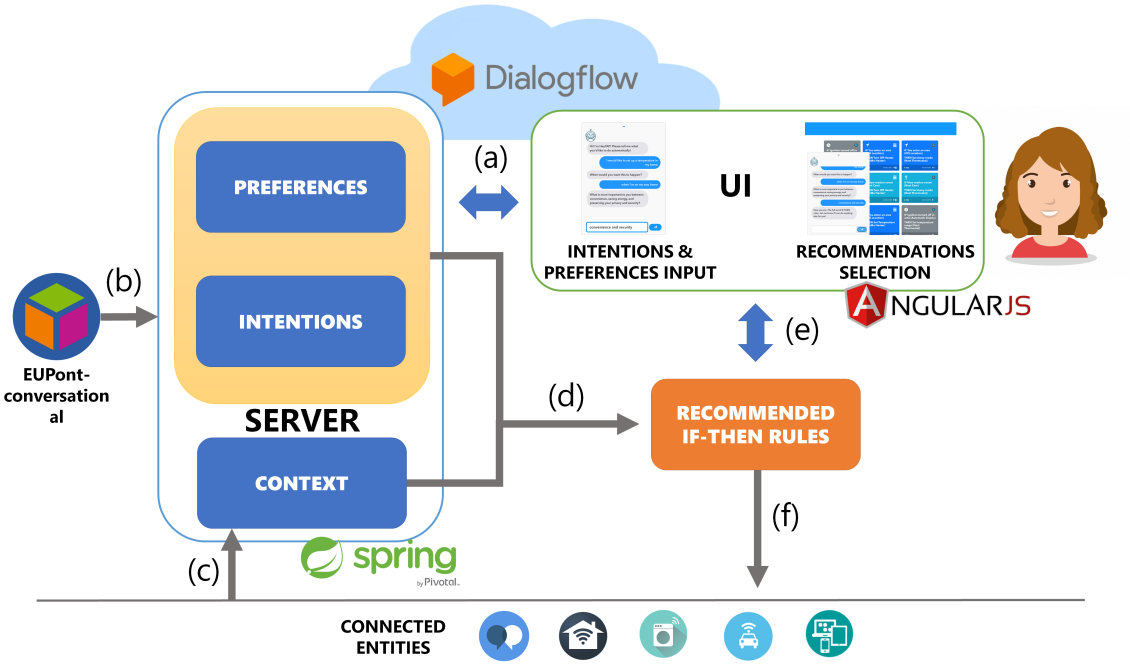


Figure 2: The architecture of *HeyTAP*. The user interacts with a conversational agent (a) to communicate to the system her personalization intentions for different contexts, along with her preferences, with different levels of abstraction. By exploiting a semantic model, i.e., EUPont-conversational (b), the server analyzes the user's input, along with contextual information related to the available connected entities (c), to infer a set of IF-THEN rules that satisfies the user's needs (d). The recommended rules are visualized in the multimodal user interface to the user, that can decide (e) which rules have to be instantiated and executed onto real smart devices and online services (f).

	Type	Functionality	Category	Entity	Technology	Where	When
Increase the air quality.	ACTION	Increase	Air quality	-	-	-	-
Open the window in the kitchen.	ACTION	Open	-	Window	-	Kitchen	-
Get a Telegram notification on my smartphone.	ACTION	Get	Communication	Smartphone	Telegram	-	-
When I'm on my way home by car.	TRIGGER	Arrive	-	Car	-	Home	-
In the evening.	TRIGGER	-	-	-	-	-	Evening
When the temperature on my home Nest thermostat drops below 20 degrees.	TRIGGER	Increase	Temperature	Thermostat	Nest	Home	-

Table 1: Some examples of action and trigger intentions that can be extracted by DialogFlow. Users can adopt different levels of abstraction by specifying one or more elements characterizing an intention.

context. As exemplified in Figure 1a, the conversational agent asks to the user if she is interested in:

- **convenience**, i.e., using all the available connected entities in an unrestricted manner;
- **sustainability**, i.e., acting on the available connected entities by trying to save energy;
- **security**, i.e., defining IF-THEN rules that preserve the user's security; and,
- **privacy**, i.e., defining IF-THEN rules that preserve the user's privacy.

The user can also ignore the request, e.g., by saying "don't mind." While different methods could be used to model the reported preferences, in our first implementation of *HeyTAP* we adopted a simple approach based on semantic filters. To filter intentions according to the described preferences, in particular, we introduced a set of OWL classes and restrictions in EUPont-conversational to automatically infer the "behavior" of the triggers and actions offered by the supported connected entities. For the sake of simplicity,

- an **energy-demanding behavior** is defined as a trigger or an action that involve a `Turn on` functionality, i.e., a behavior

that result in a new smart device that is permanently turned on unless someone (or some other rules) turns it off;
- a **privacy-invasive behavior** is defined as a trigger or an action that involve smart devices analyzing personal images, e.g., `Cameras`, or "public" online services such as `Social Networks`;
- A **security-critical behavior** is defined as a trigger or an action that involve smart devices controlling the access to physical buildings, e.g., `Doors` and *Windows*.

3.2 Recommendation

The *HeyTAP* Server uses the user's input, i.e., her intentions and preferences, along with the contextual information related to the available connected entities (Figure 2c), to infer a set of IF-THEN rules that include available and real connected entities, i.e., with IFTTT triggers and actions (Figure 2d).

By adopting a reasoning process, the server starts by analyzing the user's action and trigger intentions and extract a set of appropriate IFTTT actions and triggers, respectively. In this phase, it first extracts all the available actions, and it filters them according to available intention's elements, i.e., functionality, category, entity, technology, where, and when. The same steps are then used to extract a set of triggers, that are combined with the retrieved actions to generate a first set of IF-THEN rules. Such a set of IF-THEN rules is finally filtered by considering the user's preferences. If the user is interested in convenience, for example, such a filter has no effect. If the user is interested in preserving her privacy, instead, all the rules involving privacy-invasive behaviors are excluded. As shown in Figure 1b, the final set of recommended rules is finally visualized to the user (Figure 2e). The user can select (and complete with any additional details) one or more recommended rules involving real smart devices and online services (Figure 2f).

4 USER STUDY

To understand to what extent *HeyTAP* is able to successfully guide users from abstract needs to actual IF-THEN rules we performed an exploratory study with 8 participants. We were guided by the following research questions:

RQ1. How would users interact with *HeyTAP*?
RQ2. Is *HeyTAP* able to map abstract users' needs to executable IF-THEN rules?
RQ3. What is the users' satisfaction in using *HeyTAP*?

4.1 Participants

We recruited participants by sending emails to students enrolled in different university courses and private messages to our social circles. At the end, we involved 8 students (3 females and 5 males) with a mean age of 26 years ($SD = 1.73$, $range : 24 - 30$).

All the participants had a computer science background. On a Likert-scale from 1 (Very Low) to 5 (Very High), they stated their familiarity with the trigger-action programming approach ($M = 3.00$, $SD = 1.22$). 7 participants never used any trigger-action programming platform, while only one of them had used IFTTT a few times, sporadically.

4.2 Procedure

We devised a controlled experiment during which participants were requested to personalize a scenario by impersonating a fictional user owning a set of 24 connected entities in different contexts. The fictional user was subscribed to different online services (like Facebook and Gmail) and owned 2 smartphones. Furthermore, her home and her office were equipped with smart devices and systems, including smart doors, lights, and air conditioning systems. At the beginning of the study, we introduced participants to the trigger-action programming for personalizing connected entities, and we gave them a sheet of paper with the full list of connected entities available in the scenario, including the entity's type (e.g., lights), brand (e.g., Philips Hue), and position (e.g., ubiquitous, kitchen, office, ...). In a 15-minutes session with *HeyTAP*, participants were then free to interact with *HeyTAP* to communicate their personalization intentions and preferences. Whenever *HeyTAP* provided recommendations, we asked participants to evaluate whether the suggestions fitted their needs. At the end of the study, we performed a semi-structured debriefing session with each participant.

4.3 Measures

Since the final goal of *HeyTAP* is to suggest IF-THEN rules based on users' intentions and preferences, we defined the metrics to be collected by taking inspiration from the work of Knijnenburg et al. [17], i.e., a framework to evaluate recommender systems with a user-centric approach. According to the framweork, it is important to distinguish the following *aspects*:

- Objective System Aspects (OSA), e.g., the proposed suggestions;
- Subjective System Aspects (SSA), i.e., the users' perception of the objective system aspects;
- User Experience (EXP), i.e., users' evaluation of their interaction with the system; and,
- Interaction (INT), i.e., users' behaviors.

Table 2 describes the measures we collected during the study, with the indication of the related aspects, and the modality with which they have been collected. We use logs to evaluate the effectiveness of *HeyTAP* in addressing users' needs by means of IF-THEN recommendations (OSA), and to record the interaction (INT) between participants and *HeyTAP*. We measured, in particular, the number of exchanged messages, the number of expressed needs and weather they resulted or not in some recommendations, and the level of abstraction adopted by the participants in expressing their intentions, i.e., which of the elements described in Section 3.1.1 they specified.

Whenever *HeyTAP* provided recommendations for an expressed need, we asked participants to answer a Likert-scale question from 1 (absolutely no) to 5 (absolutely yes) to evaluate whether the provided suggestions fitted their need, i.e., the Perceived Recommendation Quality (PRQ). Furthermore, in the debriefing session, we used a Likert-scale question from 1 (absolutely no) to 5 (absolutely yes) to evaluate the Perceived Effectiveness and Fun (PEF) in using *HeyTAP*, and we asked some open questions about the perceived advantages and disadvantages of the *HeyTAP* approach.

Table 2: The measures we collected during our user study. Through different logs, we recorded the interaction (INT) between participants and *HeyTAP*, and the effectiveness of the underlying recommender algorithm (OSA). Likert-scale questions and a final debriefing session were instead used to measure Subjective System Aspects (SSA) and the User Experience (EXP) with *HeyTAP*.

Measure	Description	Collection Type	Aspect
Level of abstraction	How the user expressed her personalization intentions	Logs	INT
Messages #	Number of messages between *HeyTAP* and the user	Logs	INT
Needs #	Number of needs, i.e., intention and preferences, expressed by the user	Logs	INT
Addressed #	Number of needs addressed by *HeyTAP*, i.e., resulting in some recommendations	Logs	OSA
PRQ	The Perceived Recommendation Quality of the proposed suggestions	Likert-scale question	SSA
PEF	The Perceived Effectiveness and Fun in using *HeyTAP*	Likert-scale question	EXP
Advantages & Disadvantages	The perceived advantages and disadvantages of *HeyTAP*	Open questions	EXP

5 RESULTS AND DISCUSSION

Results are organized across our 3 research questions. First, we report on how participants interacted with *HeyTAP* (**RQ1**), i.e., which level of abstraction they adopted in their personalization intentions and which preferences they expressed. Then, we investigate the ability of *HeyTAP* in addressing users' needs (**RQ2**), and we analyze the participants' satisfaction in using our conversational agent (**RQ3**).

5.1 Interacting with *HeyTAP*

5.1.1 Intentions. The heat-map of Figure 3 provides an overview on the level of abstraction adopted by the participants to express their personalization intentions (**RQ1**).

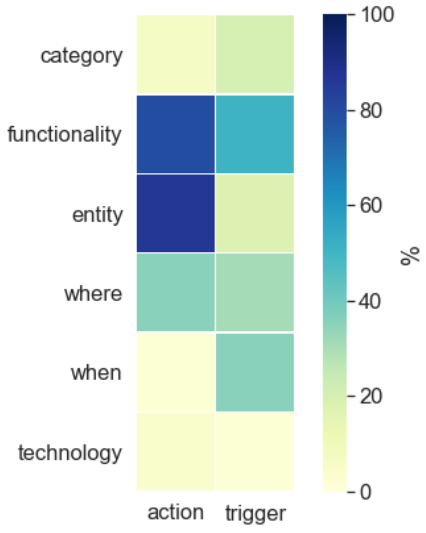


Figure 3: An heat-map characterizing how users interacted with *HeyTAP* to express their personalization intentions.

Participants rarely included *technologies*, e.g., "Philips Hue" or "Gmail," to express action (4.44%) and trigger (2.22%) intentions, thus confirming the limitations of platforms like IFTTT and Zapier [8, 20]. In line with previous works [20], in particular, trigger intentions were generally expressed in a more abstract way than action intentions. Indeed, while the 86.67% of action intentions specified an *entity* such as a door or a window, only the 17.78% of trigger intentions referred to a type of device or online service. On the contrary, trigger intentions were more likely to refer to a generic *category*, e.g., "temperature" or "communication," with respect to action intentions (20.00% *vs.* 6.67%, respectively), while action intentions included a specific *functionality*, e.g., "turn on" or "send," more often that trigger intentions (80.00% *vs.* 51.11%, respectively). Not surprisingly, a similar number of action and trigger intentions included a *where* (35.56% *vs.* 31.11%, respectively), while the *when* element was more common for trigger intentions than action intentions (35.56% *vs.* 2.22%, respectively).

5.1.2 Preferences. Table 3 reports the distribution of the preferences expressed by the participants during the study (**RQ1**).

Table 3: Distribution of the preferences elicited by *HeyTAP* during the study.

	%
Sustainability	37.93%
Convenience	24.14%
Don't mind	20.69%
Security	10.34%
Privacy	6.90%

In the majority of cases, participants expressed their preference towards sustainability (37.93%) and convenience (24.14%). In the 20.69% of cases, instead, participants did not declared any particular preference, while security and privacy were mentioned in a limited number of cases (10.34% and 6.90%, respectively).

5.2 Mapping Users' Needs

To investigate whether *HeyTAP* is able to map abstract users' needs to executable IF-THEN rules (**RQ2**), we analyzed the number of messages exchanged between participants and the tool, the number of expressed needs, and the number of needs resulting in some IF-THEN recommendations (Table 4). Each participant took advantage of her 15-minutes session with *HeyTAP* to express 4.00 needs (SD = 0.93) on average, by exchanging 28.12 messages (SD = 14.60). In 3.63 cases (SD = 0.74), *HeyTAP* was able to address the participant's need by providing some IF-THEN recommendations.

Table 4: Average results on how *HeyTAP* mapped users' needs to executable IF-THEN rules.

	M	SD
Messages #	28.12	14.60
Needs #	4.00	0.93
Addressed #	3.63	0.74

Overall, the total number of exchanged messages between participants and *HeyTAP* was 225, corresponding to 32 distinct needs and 7.03 messages on average per need ($SD = 4.61$). Of the 32 needs, *HeyTAP* successfully addressed 29 of them (90.63%). In 3 cases, only, participants were not able to get any recommendations by interacting with the tool. To understand why *HeyTAP* was not able to address such 3 needs, we analyzed the collected logs. In one case, the tool was not able to map participant's messages to the modeled elements, e.g., functionality and categories. The other 2 cases, instead, highlight an important interaction that is currently missing in *HeyTAP*. At the beginning of their usage sessions, in particular, two participants expressed their discomfort for not knowing what their connected entities could do, and used *HeyTAP* to get some recommendations:

> *"Hi! Suggest me some actions!"* (P8)
>
> *"Which services can I use?"* (P6)

As suggested by the analysis of the debriefing session (Section 5.3), knowing in advance *what* can be done could facilitate users in expressing their personalization intentions.

5.3 Participants' Satisfaction

To explore the participants' satisfaction of the participants in using *HeyTAP* (**RQ3**), we first analyzed the Perceived Recommendation Quality (PRQ) and the Perceived Effectiveness and Fun (PEF) metrics collected through the related 5-points Likert-scale questions. As reported on Table 5, participants were satisfied with the IF-THEN rules recommended by *HeyTAP* ($M = 3.93, SD = 1.20$). Furthermore, participants enjoyed using *HeyTAP* and perceived it as effective ($M = 3.75, SD = 0.83$).

Table 5: Average results on how users evaluated the Perceived Recommendation Quality (PRQ) and the Perceived Effectiveness and Fun (PEF) of *HeyTAP*.

	M	SD
PRQ	3.93	1.20
PEF	3.75	0.83

We also analyzed what participants stated during the debriefing session about the perceived advantages and disadvantages of *HeyTAP*. All the participants talked about *HeyTAP* as a useful tool to automatize users' routines. According to them, in particular, *HeyTAP* is convenient because *"it simplifies the processes needed to define automation rules"* (P7), and *"it allows the discovery of new rules from textual inputs"* (P7), especially *"for non-expert users"* (P8 and P5). Furthermore, the usage of *HeyTAP* could help users saving time and avoid possible errors, according to P4.

The majority of the perceived disadvantages were instead related to the ability of *HeyTAP* in recognizing users' sentences. P1 stated that sometimes *HeyTAP* did not immediately understand her messages, thus forcing her in rephrasing some of her requests, while P5 and P8 highlighted that the interaction with the agent was difficult when their requests were very specific. Furthermore, consistently with the results reported in Section 5.2, P6 and P8 highlighted that *HeyTAP* was not able to describe neither which devices were available nor their capabilities. Without knowing these information, they experienced difficulties in evaluating how *HeyTAP* was addressing their needs. In one case, in particular, P8 said:

> *"I was expecting some rules involving the alarm clock of my smartphone, but I do not know if this is supported."* (P8)

5.4 Limitations

The main limitation of the study is that it was exploratory in nature. In addition, this study targeted a limited number of users having a computer science background, only. A more ecologically-valid study would be to deploy *HeyTAP* in-the-wild, by testing it with different types of users. As such, our results clearly highlight the potential of the approach, and could inform follow-up studies and future development.

6 CONCLUSIONS AND FUTURE WORKS

On the one hand, contemporary trigger-action programming platforms exploit representation models that are highly technology-dependent, thus making end-user personalization of connected entities a complex task. On the other hand, the usage of a higher level of abstraction requires an effective way of selecting the real entities, triggers, and actions with which satisfying the abstract needs of the user. In this paper, we presented *HeyTAP*, a conversational and semantic-powered platform able to map abstract users' needs to executable IF-THEN rules. By exploiting a multimodal interface, users first interact with a conversational agent to communicate her personalization intentions, e.g., to program the temperature of her room, and preferences, e.g., to preserve her privacy. User's inputs, along with contextual and semantic information related to the available connected entities, are then used to extract a set of recommended IF-THEN rules, that are finally visualized to the user.

Results of an exploratory study on 8 end users preliminary confirm the effectiveness of the approach, and show that *HeyTAP* can successfully "translate" abstract users' needs into IF-THEN rules that can be instantiated and executed by contemporary trigger-action programming platforms. In future works, we will extend *HeyTAP* with the suggestions we extracted from the participants of the exploratory study, e.g., by adding the possibility of asking the conversational agent which connected entities can be personalized and which capabilities do they offer. Furthermore, we are also investigating how to include in *HeyTAP* more complex rules, e.g., by supporting multiple actions, and we are planning a more ecologically-valid study that involves the in-the-wild deployment of the tool.

REFERENCES

[1] 2020. Amazon Alexa. https://developer.amazon.com/en-US/alexa Accessed: 2020-01-20.

[2] 2020. Google Assistant. https://developers.google.com/assistant Accessed: 2020-01-20.

[3] 2020. IFTTT. https://ifttt.com/ Accessed: 2020-01-20.

[4] 2020. Zapier. https://zapier.com/ Accessed: 2020-01-20.

[5] B. R. Barricelli and S. Valtolina. 2015. *End-User Development: 5th International Symposium, IS-EUD 2015, Madrid, Spain, May 26-29, 2015. Proceedings.* Springer International Publishing, Cham, Germany, Chapter Designing for End-User Development in the Internet of Things, 9–24. https://doi.org/10.1007/978-3-319-18425-8_2

[6] Vint Cerf and Max Senges. 2016. Taking the Internet to the Next Physical Level. *IEEE Computer* 49, 2 (Feb 2016), 80–86. https://doi.org/10.1109/MC.2016.51

[7] F. Corno, L. De Russis, and A. Monge Roffarello. 2017. A Semantic Web Approach to Simplifying Trigger-Action Programming in the IoT. *Computer* 50, 11 (2017), 18–24. https://doi.org/10.1109/MC.2017.4041355

[8] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019. A high-level semantic approach to End-User Development in the Internet of Things. *International Journal of Human-Computer Studies* 125 (2019), 41 – 54. https://doi.org/10.1016/j.ijhcs.2018.12.008

[9] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. 2019. RecRules: Recommending IF-THEN Rules for End-User Development. *ACM Trans. Intell. Syst. Technol.* 10, 5, Article 58 (Sept. 2019), 27 pages. https://doi.org/10.1145/3344211

[10] Giuseppe Desolda, Carmelo Ardito, and Maristella Matera. 2017. Empowering End Users to Customize Their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools. *ACM Transaction on Computer-Human Interaction (TOCHI)* 24, 2, Article 12 (April 2017), 52 pages. https://doi.org/10.1145/3057859

[11] Anind K. Dey, Timothy Sohn, Sara Streng, and Justin Kodama. 2006. iCAP: Interactive Prototyping of Context-aware Applications. In *Proceedings of the 4th International Conference on Pervasive Computing* (Dublin, Ireland) *(PERVASIVE'06)*. Springer-Verlag, Berlin, Heidelberg, 254–271. https://doi.org/10.1007/11748625_16

[12] Mathias Funk, L.-L Chen, S.-W Yang, and Y.-K Chen. 2018. Addressing the need to capture scenarios, intentions and preferences: Interactive intentional programming in the smart home. *International Journal of Design* 12 (04 2018), 53–66.

[13] Giuseppe Ghiani, Marco Manca, Fabio Paternò, and Carmen Santoro. 2017. Personalization of Context-Dependent Applications Through Trigger-Action Rules. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 2, Article 14 (April 2017), 33 pages. https://doi.org/10.1145/3057861

[14] Will Haines, Melinda Gervasio, Aaron Spaulding, and Bart Peintner. 2010. Recommendations for End-User Development. In *Proceedings of the ACM RecSys 2010 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI)*.

[15] J. Huang and M. Cakmak. 2015. Supporting Mental Model Accuracy in Trigger-action Programming. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan) *(UbiComp '15)*. ACM, New York, NY, USA, 215–225. https://doi.org/10.1145/2750858.2805830

[16] Ting-Hao K. Huang, A. Azaria, and J. P. Bigham. 2016. : Creating IF-THEN Rules via Conversations with the Crowd. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Santa Clara, California, USA) *(CHI EA '16)*. ACM, New York, NY, USA, 1555–1562. https://doi.org/10.1145/2851581.2892502

[17] Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. 2012. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction* 22, 4 (01 Oct 2012), 441–504. https://doi.org/10.1007/s11257-011-9118-4

[18] Xianghang Mi, Feng Qian, Ying Zhang, and XiaoFeng Wang. 2017. An Empirical Characterization of IFTTT: Ecosystem, Usage, and Performance. In *Proceedings of the 2017 Internet Measurement Conference* (London, United Kingdom) *(IMC '17)*. ACM, New York, NY, USA, 398–404. https://doi.org/10.1145/3131365.3131369

[19] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, and Michael L. Littman. 2014. Practical Trigger-action Programming in the Smart Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. ACM, New York, NY, USA, 803–812.

[20] B. Ur, M. Pak Yong Ho, S. Brawner, J. Lee, S. Mennicken, N. Picard, D. Schulze, and M. L. Littman. 2016. Trigger-Action Programming in the Wild: An Analysis of 200,000 IFTTT Recipes. In *Proceedings of the 34rd Annual ACM Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3227–3231. https://doi.org/10.1145/2858036.2858556

[21] Lina Yao, Quan Z. Sheng, Anne H.H. Ngu, Helen Ashman, and Xue Li. 2014. Exploring Recommendations in Internet of Things. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval* (Gold Coast, Queensland, Australia) *(SIGIR '14)*. ACM, New York, NY, USA, 855–858. https://doi.org/10.1145/2600428.2609458