A Mobile Augmented Reality App for Creating, Controlling, Recommending Automations in Smart Homes

ANDREA MATTIOLI, CNR-ISTI and University of Pisa, Italy
FABIO PATERNÒ, CNR-ISTI, Italy

Automations in the context of smart homes have been adopted more and more frequently; thus, users should be able to control them and create automations most suitable to their needs. Current solutions for this purpose are based on visual apps with conceptual representations of possible automation elements. However, they tend to be static, abstract, and detached from the user's real context. In this paper, we propose a novel solution based on mobile augmented reality, which provides situated, dynamic representations associated with the physical objects available in the current users' context while they are freely moving about. It allows direct interaction with the objects of interest, monitoring nearby objects' automations while moving, and creating new automations or modifying existing ones. It also supports users with recommendations of object and service configurations relevant to complete the editing of the new automations. The paper also reports on a user test, which provided positive feedback.

CCS Concepts: • **Human-centered computing → Mixed / augmented reality; Ubiquitous and mobile devices; • Information systems → Recommender systems**

KEYWORDS: Augmented Reality, Internet of Things, Trigger-Action Programming, End-User Development, Recommender Systems

ACM Reference format:

Andrea Mattioli and Fabio Paternò. 2023. A Mobile Augmented Reality App for Creating, Controlling, Recommending Automations in Smart Homes. In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 7, MHCI, Article 195 (September 2023), 22 pages, https://doi.org/10.1145/33XXXXX

1 INTRODUCTION

The number of connected objects and sensors is steadily increasing, and they are permeating all areas of our life, thus enabling various types of smart spaces. In such spaces, various automations are made possible by involving the behaviour of connected objects, devices, and services. Such emerging technological settings open up great opportunities and new possibilities, but they also introduce new risks and problems. For example, the automations can generate unwanted effects or people may have difficulties in understanding the generated automations (see for example [49]), thus it becomes important to provide tools that allow users

Author's addresses: Andrea Mattioli, CNR-ISTI, Pisa, Italy, andrea.mattioli@isti.cnr.it; Fabio Paternò, CNR-ISTI, Pisa, Italy, fabio.paterno@isti.cnr.it.

2573-0142/2023/06 – Article#195 $15.00
https://doi.org/124564XXXX

to control and configure smart environments consisting of even hundreds of interconnected devices, objects, and appliances in their daily environments, such as homes, offices, industries.

End-User Development (EUD) [27, 31] aims to propose methods, techniques, and tools that allow people who are not professional designers to create or modify a software system. Several proposals have been put forward for the smart home domain in the research area of EUD tools, covering various objectives, needs and approaches, e.g. [6, 7, 13]. One EUD approach that seems particularly relevant for end users of smart spaces is that based on Trigger-Action Programming (TAP), which does not require particular algorithmic abilities. Examples of its application to the smart home domain are reported in [30, 37]. They are based on rules connecting situations or events, which are relevant to users and are derived from sensors or services, and actions indicating what specific objects or applications should do.

Various composition paradigms have been proposed for representing the relevant concepts and supporting the development process of such rules. Usually, they are based on the visual channel and differentiate in terms of the presentation and interaction styles. A common approach is to support some kind of visual wizard aiming to guide the users by limiting their possible selections (e.g., IFTTT[1], EFESTO [16], TAREME [20], ImAtHome [18]). In general, visual editors with conceptual representations of possible automation elements tend to be large, comprehensive, static, abstract EUD tools detached from the user's real context. It may not be straightforward how to navigate the large number of elements, and understanding what real element they refer to may sometimes require technical knowledge. One possible way to address such issues is to use a conversational style exploiting chatbots ([19, 26]). However, if such chatbots use textual interaction the composition process may be quite long, while vocal interfaces may easily misunderstand user requests. Thus, there is a need for more narrowed, situated, dynamic representations associated with the physical objects available in the current users' context while they are freely moving about.

In this perspective, Augmented Reality (AR) can play a useful role. However, AR dedicated headsets are still expensive and bulky and can also be problematic for end users to select and configure these devices [2]. A solution exploiting the smartphone can instead potentially be largely adopted since people usually have such a device with them. It opens up the possibility of direct interaction with the objects of interest, monitoring nearby automations that involve an object while moving about, creating new automations or modifying existing ones. This solution can be valuable to support users who may not have programming experience in creating, editing, and viewing automations in an easier, faster, and more engaging way. Furthermore, showing which automations are currently active on nearby objects, it increases the overall transparency of the smart home. Pioneering work in the context of mobile applications looked into context-dependent automatic actions, introducing and assessing a tool for allowing users to select conditions for context-based behaviour of applications [32] and evaluating how users' perceived sense of control in these applications can impact the user experience [4]. However, these studies referred to mobile devices with limited capabilities and without considering augmented reality techniques.

A further possibility is to also support users with recommendations of relevant items in the editing of new automations. Considering the wide search space of triggers and actions combinations and the domain knowledge required to write TAP rules [50], introducing a recommendation system (RS) to provide support can be an effective way to make this process

[1] https://ifttt.com/explore, last accessed 16/06/2023

less challenging for users because it could help users discover new potential automation. In this case, AR can relieve the problem of displaying too much information on a smartphone's screen's (limited) area, moving information into the real space over the objects they refer to. Current approaches to AR in IoT (e.g. [25, 46]) are mostly limited to providing users with the possibility to interact and control single connected objects. There have been few attempts to explore the use of AR for creating or modifying simple automations in daily environments. One example is SAC [1], but it has limited usability and has not the possibility to support users also with recommendations of possible relevant automations.

Looking at current solutions for EUD for smart home automation, it emerges that user control of automations in such settings is still limited, in particular because the selection of the connected objects and the automation configuration can be challenging. Investigating a mobile AR solution to allow users to edit even compound automations, also with the support of recommendations of configurations for relevant objects and services is a promising direction to improve the corresponding user experience. This research objective concretizes in the following research questions: (RQ1) How to design an augmented-reality based EUD solution to allow users to easily create automations that can involve multiple triggers and actions, (RQ2) How to exploit AR to identify the active automations and understand which devices they are associated with, (RQ3) How to introduce the possibility to provide recommendations of possible automations in such AR platform, and (RQ4) Assessing the introduction of recommendation support in the AR tailoring tool in order to facilitate the specification of the desired ones.

2 RELATED WORK

2.1 AR to Configure Smart Homes

Previous work [35] indicated that users are eager to benefit from on-demand information, assistance, enhanced sensory perception, and play offered by AR across many locations at home. Research to identify in concrete how to provide such support has followed several directions. One of the first contributions exploring the possibility of using AR not only to control but also to connect the behaviour of different IoT objects is Reality Editor [23]. Using AR, Reality Editor maps graphical elements directly on top of tangible interfaces associated with physical objects, such as push buttons or knobs. By connecting tags of different objects (by drawing a line between them), the user can program multi-object functionality. Thus, its goal is to provide additional possibilities for user interaction with the functionalities of available objects and devices, while we aim to support users in flexibly specifying automations involving multiple such objects. HoloFlows [38] is aimed at making simpler the configuration and modelling of IoT workflows through a no-code Augmented/Mixed Reality approach. It exploits concepts from the Business Process Modelling (BPM) domain and allows the definition of automated tasks involving one or more IoT devices. End users can use “virtual wires” to connect physical IoT devices and create processes involving them. It adopted the optical see-through approach to display holographic images on the glasses and to position them on the scene. The authors put forward an extended contribution [39] where the HoloFlows modelling approach is compared with a classical BPM approach (implemented in the Camunda tool) and with a flow-based approach (NodeRED). In contrast with our work, HoloFlows requires dedicated hardware, and it is more oriented toward the “wiring” of objects physically nearby. MagiPlay [42] is a serious game targeted at children that exploits AR. Its main goal is to support the learning of concepts related to computational thinking through an engaging augmented environment where young learners can combine 3D visualizations to generate IoT automations. Unlike our work, MagiPlay does not aim to support several aspects concerning the creation, editing and exploration of everyday automations in smart homes, since its goal (learning) and target users (children) are different. ARticulate [9] supports a method using Snapchat-like contextual photo messages enhanced by two technologies (AR and autocomplete) to allow users to determine available functionality and achieve their goals in one attempt with a smart space they have never seen before. It provides useful support but unlike the proposed solution does not consider the automations involving multiple connected objects, and it is a tool aimed at only discovering functionalities, not configuring them. SAC [1] was a first attempt to support users in controlling automations but differently from the proposed work it required them to be very close to the objects to frame them and trigger the associated visualisations, and it only supported the creation of single trigger-single action rules, without any recommendation support.

2.2 TAP & Recommendations

Automation recommendations in smart home settings are usually generated from two different sources of information: an analysis of the user behaviour detected through the support of sensors, devices, and applications; or sets of previously created and used automations. RuleSelector [41] is an example of the former approach: it considers the user behaviour (but only the part that can be detected through the smartphone) to identify the possible contexts of use and the actions that the user performs when they occur and suggests the corresponding automation according to metrics such as confidence, contextual specificity, interval count, and total action coverage. Another approach is Trace2TAP [52]: it automatically synthesises TAP rules from traces of time-stamped logs of sensor readings and manual actuation of devices. It was tested in seven offices, and the detected traces contained 18 unique actions on 9 lamps that were considered targets for automation. Thus, also this approach was limited in terms of the possible rules that could be inferred. An example of an approach of recommendations based on previously created ones is RecRules [10]. It exploits the EUPont ontology and a knowledge graph of collaborative information between users and rules, adapting suggestions to the user's high-level intention during the rule composition phase. HeyTap [11] is a conversational interface also leveraging EUPont, intending to facilitate the configuration of devices and services introducing an RS in a conversational platform. Another semantic approach to recommendations is rtar [47], where a knowledge graph that also includes context-aware information is used to recommend TAP rules. The approach is used on a hybrid model that contains a rule collaboration graph and a functionality hierarchy. A ranking model is trained on the knowledge graph to sort the candidates according to their relevance. A different approach has been considered in RecipeGen [50], which generates TAP rules from natural language descriptions. It leverages a Transformer sequence-to-sequence (seq2seq) architecture, using pre-trained models to warm-start the encoder in the seq2seq. The approach formulates the user behaviour as sequential, that is users select a trigger first and then specify an action. All these approaches leverage the single-trigger single-action IFTTT datasets, hence without allowing multiple triggers/actions automations, and only RecipeGen and HeyTap can consider further data such as the configuration values inserted by users. To our knowledge, no study has investigated the use of an RS for a smart home automation AR-based platform.

3 DESIGN OF THE PROPOSED SOLUTION

3.1 Requirements

Based on the literature review and our experiences with previous tools for TAP, we identified a list of main requirements for the solution.

R1 Remove the need for a detection phase before interacting with the environment. In previous work [1], the object recognition process received mixed feedback. Some users indicated that the recognition was not always quick enough. Furthermore, the need to go near each object can be a nuisance, especially for people with mobility problems. Users should hence be able to enter a smart space and invoke the desired functionality without the need to go near an object and wait for the detection.

R2 Allow for more expressive rules. It should be possible to select multiple triggers and actions, in the order preferred by the user. Many useful behaviours cannot be expressed using a single trigger – single action structure [5]. The proposed prototype supports the Event-Condition-Action (ECA) definition, as it has shown to be suitable to describe IoT automations [3, 17, 33], and it allows users to express their personalization needs [15]. A specific order for the selection of the rule elements has not been imposed, because order flexibility has a positive impact on users' performance and satisfaction [16] and from our previous experiences we did not see a clear preference for the trigger - action order or vice versa. Using AR, people are even more stimulated to create automations without following the pre-established ECA order, since they can move freely about in the environment. Differently from most previous work (e.g. [1]) that allows the creation of simple "single trigger-single action" automations, the prototype supports the possibility of multiple triggers and multiple actions: once the configuration of a rule element has been completed, the user can continue with the configuration of another object/service or save the rule if the conditions are met.

R3 Provide visualisations that support user control and understanding of the environment's automations. AR can be used to map graphical elements directly over the physical objects they refer to and control them [23]. These visualisations can also be used to provide discoverability of the environment, for example by making explicit the automations present in an unfamiliar space [9]. This feature is also useful in a domestic environment, as with multiple users, or when there are seldom used automations that can be forgotten. For this reason, it should be easy to perceive if an automation is present/can be defined on an object, to understand what triggers and what objects are involved, and how to modify these automations.

R4 Provide rule recommendations that match the possible user intents in the rule creation phase. The large number of smart objects and services available makes the possible ways to combine them always growing. The creation of an IoT automation is not a "one-shot" operation, but it consists of multiple selection and configuration steps [12]. This calls for some kind of recommendation support, in particular with devices with limited screen size, to help users by providing them with relevant objects or services configurations during these various phases. For this reason, a RS specific to this setting should be introduced. The advantage of using AR is that the recommendations are generated directly above the objects they refer to, addressing the limitation of having little screen space and the need for the user to make the object-representation mental link.

R5 Allow the selection of external services to be accessed in relation to home objects. Automation rules are not only about connected objects. Previous analysis of user-created automations [37, 44, 45] indicates that services such as weather and notifications were commonly used as triggers and actions. In general, it should be expected that users may need to express behaviours that mix capabilities directly related to objects with others depending on external services. The application should hence give users access to both of these types of capabilities.

R6 Combine 3D abstract representations and 2D mobile interfaces. In AR for the smart home context, where the fidelity of the virtual representation to the real objects is not essential, it is possible to manage the interaction using graphical representations more connected to real objects [23] or more abstract ones [53]. In a preliminary version we used 3D objects resembling the real ones and UI elements integrated into the augmented view, but a preliminary lab evaluation of this solution (3 participants) indicated that the interface became too crowded. Furthermore, for scalability reasons it is problematic to provide representations for each object in the home. Hence, a more balanced approach using 3D abstract representations together with form-based interfaces and icons that appear only when needed[2] seems more promising.

3.2 Application Implementation and Interaction

The proposed solution is an Android application implemented using Unity and the ARFoundation library. The main functionalities of the application are “Create automation”, “Edit automation”, “Explore environment”, and “Get objects position”. Within the functionalities, users are presented with 3D AR visualisations, 2D panels, and icons (R6). The “Create automation” functionality allows users to generate automations while roaming freely about in the environment, selecting the visualisations associated with smart objects and configuring each “rule element” relevant to the behaviour they want to determine (R2). There is no need to go near an object to initiate the object recognition and then interact with it since the application knows the position of the objects. This feature (R1) is implemented by associating an AR anchor to a location for each activatable object in the smart home. This step can be performed by the user at the first use of the application, and the visualisation may be updated when there is a change of position.

Different graphical elements are used to visualise some aspects of the automation composition. For instance, an "inactive exclamation mark”, with a blue colour, indicates that an object has some features that can be used in the rule configuration, while an “active exclamation mark”, coloured in green, signals that one of these features is selected and configured in the automation that the user is currently editing. We selected this shape and material (semi-transparent, see Figure 1) because we needed the visualization to be easily identifiable and selectable. Still, at the same time, it should occlude as little as possible the scene behind it. We chose to use visual feedback (active and inactive) to convey whether an object is used in a rule because it is crucial information not to confuse users during the creation of automations.

Each object discloses one or more functionalities that can concern sensing its state (and hence be used as triggers in the automation) or performing some change in it (and so be used as actions).To address the various possible user needs it is also possible to compose automations that involve services that are not related to a specific object (R5), such as “send a reminder”, “training time”, or “weather forecast”. After the user has inserted a rule element, a call to the RS obtains some configuration recommendations that can be relevant to what the user has inserted (R4). These recommendations are displayed using some “info panels” placed over the objects they refer to. Users can visualise and eventually modify the already created rules using the “Edit automation” application functionality (R3). The "Explore environment" functionality is used to

[2] https://www.nngroup.com/articles/ar-ux-guidelines/, last accessed 16/06/2023

visualise all the automations referring to the nearby objects. The goal is to make perceivable the relation between automations and objects.

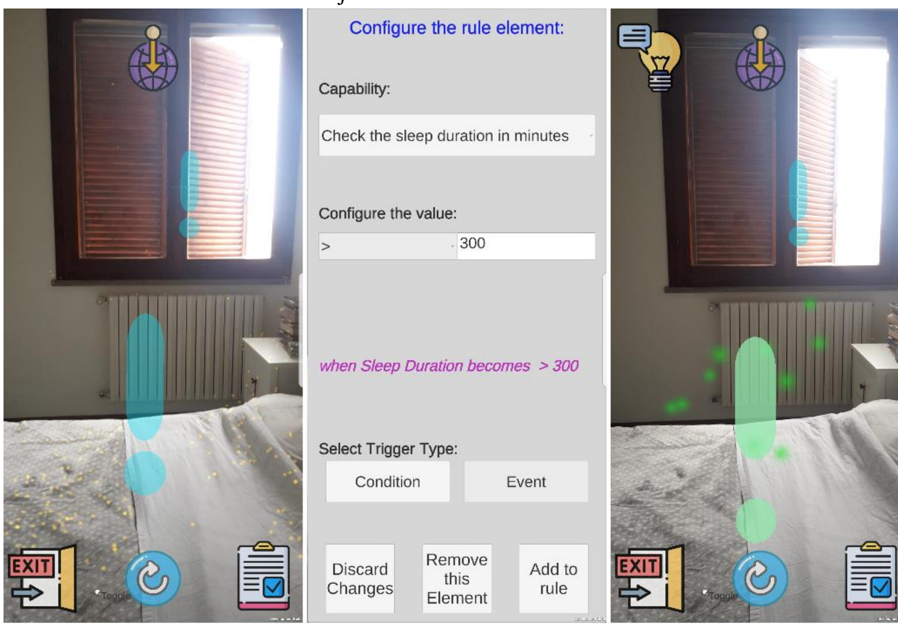

Fig. 1. (Left) the "Create automation" function has just been selected (all the "exclamation marks" are inactive); tapping on a mark opens the configuration panel (centre); and then the related "exclamation mark" is activated (right).

After selecting "Create automation", all the "exclamation marks" in the environment are inactive (Figure 1, left). Tapping on an inactive (blue) exclamation mark loads the panel for configuring the rule element (Figure 1, centre). For instance, the bed sensor panel allows for selecting between the "Sleep duration in minutes" and "Bed occupancy" capabilities, and then configuring the rule elements (whether "event" or "condition" trigger type, the value, and the operator if needed). At this moment, the user is also presented with the choice between "event" and "condition" trigger types. An event refers to the moment a device or service changes its state. A condition is a statement that can be evaluated as true or false. Huang and Cakmak noted (and recent studies [51] confirmed) that not communicating this distinction might generate ambiguities and that it is important that the system provides both state and event triggers for the same functionality [24]. They also suggest terms to clarify this distinction. We adopted the "when (event) if (condition)" terminology, as the term "when" couples with the idea of an exact, punctual moment in which the change of state occurs, and the term "if" maps to the Boolean condition to be checked. After the user has configured a rule element, the corresponding "exclamation mark" is activated (Figure 1, right), signalled by a particle effect, and by changing its colour to green. An active "exclamation mark" can be selected to modify or delete the rule element, or to add another functionality from the same object to the automation. For instance, both the "Temperature level" and "Humidity level" can be selected from a multipurpose sensor. The "recommendation info panel" (Figure 2, left) shows natural language configurations of objects relevant to complete the rule fragment. They are placed over the related object. Similarly, in the "Explore environment" mode the "rule info panels" (Figure 2, right) display natural language descriptions of the automations over the corresponding objects.

After a rule element is configured, the "Light bulb" icon will appear in the upper-left part of the screen. Tapping the bulb shows a panel with the full list of recommendations. It includes those related to services which cannot be displayed within the "recommendation info panel" because not associated with an object. The "globe" icon in the upper-centre part of the screen displays services that are not directly related to an object. The "floppy" icon will appear on the upper-right part of the screen when the completeness conditions are met (at least one element for the trigger and one for the action), allowing the automation to be saved. To support users during the creation of an automation, a "rule display" icon on the lower right part of the screen is selectable (see Figure 2, centre). Tapping this icon shows the rule currently in creation in terms of the event-condition-action scheme.

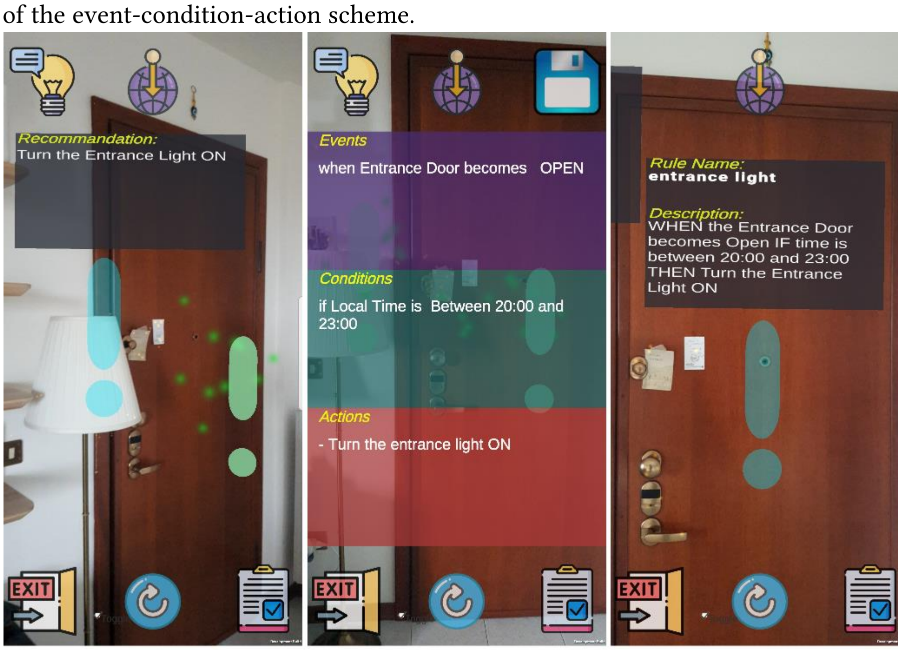

Fig. 2. (Left) after configuring the trigger "Entrance door", a recommendation is placed over the Entrance light; (centre) a schematic view of the rule currently in editing: (right) the "Explore environment" modality displays the created automations over the associated virtual objects.

By way of a concrete example, the following describes the configuration of an automation. The user wants to automate the air circulation in the bedroom. She wants the bedroom window to open at 9:00 am, but only if she has already got up and the weather is good. She also wants to receive a notification when this automation is activated. She starts the configuration from the "date and time – current" service. So, she taps on the "globe" icon and selects the service. As this rule element is the "trigger point" of the automation, she chooses the "event" trigger type. Then, she inserts the desired operator ("equal", "more than", "less than", or "between") the time, and finally taps the "add to rule" button. The screen switches back to the camera view", and an "info panel" with a recommendation for configuring the rule element "if bed occupancy is true" appears over the bed. This suggestion is shown since the "bed occupancy" element is often present in rules with the "current time" triggers indicating similar hours. So, she taps on the "light bulb" icon, selects this recommendation, changes the bed occupancy value to false and adds it to the rule. She continues the configuration of the automation by selecting the "weather – weather condition" option from the services list and configuring it. Then, she repeats this operation for the "reminder" service. Since she added the phrase "I opened the bedroom window!" in the text part of the reminder, the recommender system places a related "info panel" over the bedroom window, suggesting the action "Open bedroom window". Hence, she selects this recommendation as before. Alternatively, she can choose the "exclamation mark" in front of the window and configure this rule element. Finally, she concludes by tapping the "floppy" icon to save the automation.

4 RULE ELEMENTS RECOMMENDATIONS

Our aim is to obtain a RS able to propose rule elements relevant to what the user has already entered. We focused on providing rule fragments instead of full rule recommendations because in previous work (for example, [34]) it was observed that users tend to prefer some kind of guiding recommendation, e.g., "Which object can I use together with this one?". This also resulted in a better integration with the application, because a rule fragment can be selected and directly added to the rule currently in editing. The RS should consider the specific characteristics of the AR setting. First, the creation of rules does not necessarily occur in a mainly sequential manner. In this approach, the users can move freely in an environment, and the selection of the desired rule element can also be made in an exploratory way. Hence, a different design solution is necessary. Another aspect to consider is the scarcity of space on the mobile screen, and that users prefer to examine only a short list of recommendations [41]. For these reasons, we opted for a solution that aims to maximise the use of a small number of automation fragments to suggest after at least one rule element has been configured.

We used a publicly available trigger-action rules dataset[3] to train and evaluate the model. We used this instead of an IFTTT dataset because we wanted to let users express more flexible automations, potentially composed of multiple triggers and actions, which are often necessary to meet their needs. The IFTTT datasets instead contain only simple one-trigger one-action rules. The dataset used contains 434 rules, with 166 different rule element classes. A class is composed of the name of the service or object capability, such as "lamp colour" or "outdoor temperature", and its parent element, which is either a room or an object category. For instance, the functionalities regarding sleep duration and bed occupancy are grouped under the parent category "bed", whilst the sensing and action functionalities concerning doors and windows are classified using the corresponding room name.

4.1 Recommendations Generation

The algorithm aims to provide recommendations based on a similarity measure (how well the recommendation candidates match the user query) balanced with a diversity measure (how heterogeneous the entire recommendation list to present is). We used the Doc2Vec model to generate the similarity scores for a recommendation candidate. Doc2Vec [29] is an extension of the Word2Vec model aimed at learning document-level embeddings [28]. It can be applied to automation rules because these rules often have a textual description inserted by the creator, or it can be generated from the rule components if it is not present. Doc2vec captures the semantic

[3] https://github.com/andrematt/trigger_action_rules, last accessed 16/06/2023

relation between the rule fragment in input and the other automations in the dataset by analysing the context of the words in the automation description. This allows the abstraction from specific terms such as the device brand or service. Furthermore, using the full text of an automation unlocks the information that resides in the user-defined part of the automation, e.g., the text of a reminder, which enables linking the reminder "close the microwave door" to the related object and action. The diversity metric is the inverse of the result of a similarity query between the latent semantic indexing [14] of the candidate rule element and the latent semantic indexing of the concatenation of the other rule elements already in the recommendation list. The recommendations model was trained using the Gensim library in Python [36] and it is served by a Flask application (see Figure 3).

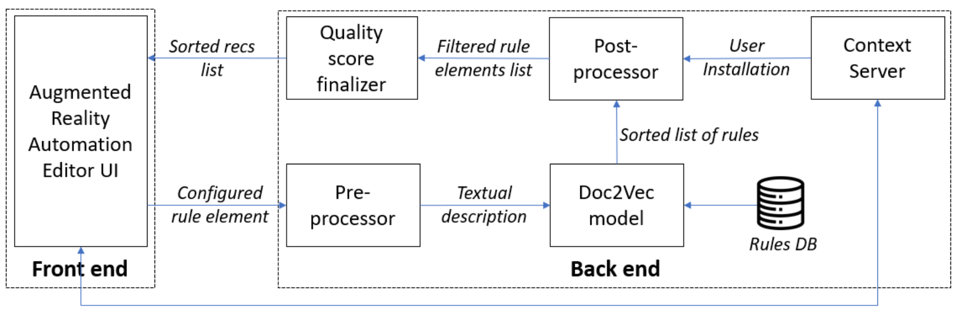

Fig. 3. The recommender system architecture and dataflow of the recommendation generation.

More in detail, the generation of a recommendation occurs as follows. After the user configures a rule element, the **Pre-processor** generates its corresponding textual description. This module reconstructs a quasi-natural language structure starting from the various configured rule parts, then tokenized and stemmed. The **Doc2Vec model** module then makes a similarity query between this representation of the user input and the model (previously trained with the natural language description of rules obtained from the public dataset introduced before). The similarity is hence used as a measure of relevance. The output is a sorted list of rules. The **Post-processor** module extracts the constituent elements of these rules until the required number of rule elements is reached (e.g., 4N, where N is the number of recommendations to show). Rules elements coming from rules that cannot be activated in the user installation are discarded based on the information provided by the **Context Server**. Rule elements that correspond to the one inserted by the user are also discarded. The resulting filtered list is passed to the **Quality Score Finalizer**. This component processes the received rule elements, selecting the candidate that maximises the formula W * similarity + (1-W) * diversity [40]. In the formula, all the parameters vary between 0 and 1, and W is the weight assigned to balance the relevance and diversity parts. The algorithm stops when the final recommendation list reaches length N.

4.2 Recommendations Evaluation

We performed some checks to make sure that the provided recommendations are relevant to what the participants could insert during the user test. We trained the Doc2Vec model (300 epochs, vector size = 100, minimum word count = 1) with the complete dataset (434 input phrases tokenized and stemmed). Then, we passed each training data as input to the trained model to assess whether it can recognize as most similar the training phrases themselves, as suggested in the documentation[4]. Although not a real accuracy value, the score indicates whether the model is behaving in a consistent and expected manner. We found that in 95.62% of cases, the input is returned within the top 3 positions of the output, meaning that the model is consistent with the data. Then, we assessed the relevance of recommendations using a greedy algorithm as a baseline. The greedy algorithm works by selecting at each step the candidate rule element that maximises a metric, in this case a support score, obtained by counting how many times a rule element appears in rules together with the one selected by the user. Therefore, unlike the algorithm based on Doc2Vec, which through the natural language description knows all the components of a rule element, the greedy algorithm only uses the class of the rule element (e.g., "weather-is_raining"). To simulate the user input, we modified the dataset by assigning a row for each rule element (instead of a row for each rule) and passed each row to the RS to obtain five recommendations.

We used the Hit rate @5 metric [21] to assess the performances, i.e., counting as a positive score when these recommendations contained a rule element present in the original input rule. The Doc2Vec-based algorithm obtained a score of 0.793, while the greedy a score of 0.493. We repeated the assessment with 10 random train/test splits of the dataset (80% training) to assess the model performance with unseen data. We obtained an average score of 0.580 for Doc2Vec and 0.374 for the Greedy algorithm. Note that these scores are influenced by the small size and sparseness of the dataset, meaning that many rule elements are present only one or a few times. Using only the rules related to the most common devices and services (the ones also used in the user test environment, see section User Test) gives higher scores (0.717 for Doc2Vec, 0.408 for Greedy).

Table 1. Changing in the Hit Rate and Diversity scores at the change of candidate rule elements number.

Candidates	1N	2N	3N	4N	5N	6N	7N	8N
Hit rate	0.733	0.725	0.683	0.675	0.633	0.608	0.542	0.492
Diversity	1120.526	1341.227	1412.714	1467.887	1490.088	1505.569	1515.127	1519.098

Finally, we examined how to balance the number of candidate rule elements to be extracted from the rules most similar to the user input. For this test we used the Doc2Vec model and the dataset from one of the rounds with the user installation rules, fixing the W parameter to 0.5. We passed all the test dataset rows to the algorithm, calculating for each output (the list of 5 recommendations) a diversity score for the whole recommendation list. This score is the sum of the cosine distance between the latent semantic indices of all the pairs of phrases in the result list. Table 1 shows a comparison of the diversity (aggregated for all the input queries from the test dataset) and hit rate results, obtained from different coefficients for N (N = number of recommendations to show, in this case 5). It can be observed that 1N gives the best results from the hit rate point of view, while assigning to N a value between 2N and 4N gives good results both from the accuracy and diversity perspectives.

5 USER STUDY

[4] https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html, last accessed 16/06/2023

The user study aimed to assess the approach with respect to the main themes identified in the research questions. Fifteen participants (6 females) were recruited through mailing lists or via email. Their ages ranged from 26 to 38 years (average = 31.6875, std. dev. = 3.86). Eleven participants had a bachelor's degree or a higher title. Nine had no programming experience, and none of them reported a professional level of proficiency with programming. They reported good interest in new technologies (4 on average on a scale between 1 and 5). Twelve had some experience with AR (the most reported answer was Pokémon Go reported 6 times). One of them had some experience with TAP (she used IFTTT for some simple automations involving services), but none had experience with smart homes. A short document with an introduction to TAP and AR Rule Editor was sent to those who accepted to participate. They had to go to the apartment made available for the experiment.

5.1 Test organisation

The smart home installation was done in a flat made available by a volunteer and consisted of 15 connected objects deployed in 4 rooms (see Figure 4). In addition, seven services not associated with objects were selectable in the application, namely reminders, alarms, current weather (which includes the "check if it is raining", "check if it is snowing", and "check the outdoor conditions" functionalities), twenty-four hour weather forecast (including the same three triggers of the "current weather" service), date-time (including the "hour of the day, day of the week", and "day type"), relative position, and training (which includes the "daily steps" and "training time").

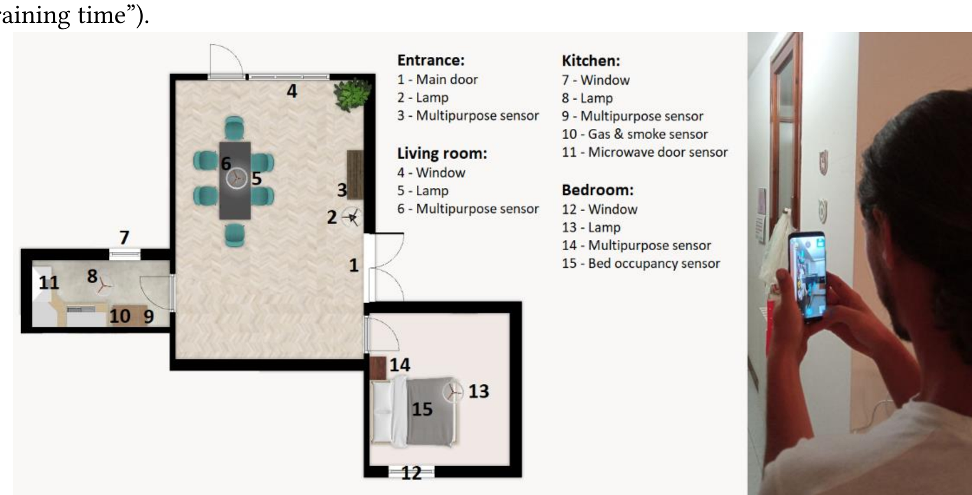

Fig. 4. (Left) plan of the smart home installation used for the user test; (right) a test participant while carrying out the tasks.

The test session was organised in three phases: introduction and familiarisation, task performance, and final questionnaire. In the first phase, we briefly illustrated the main functionalities of the application, and the different environments and objects present in the smart home installation. Then, participants were free to familiarise themselves with the application (installed on a Samsung Galaxy S8) and to create some test automations. In the second phase, they had to carry out the tasks detailed in the following. After the tasks, a questionnaire was administered. The first part was about the participants' demographic, while the remaining questions concerned specific aspects of the application, gathering feedback on positive/negative aspects, and how to improve it. They were free to fill out the online questionnaire on the spot, or later at home. After the test, many participants spontaneously remained to discuss their experience with the application, thus providing further feedback.

The user study was performed as a within-subjects study where each participant tested both conditions (with and without recommendation support) because of the difficulties in bringing numerous users into the available flat. Two smart home scenarios were presented to participants. For each scenario, they had to generate four automations, each one corresponding to a task. Two were simple ones (one trigger and one action) and two compounds (the composition involved either the trigger or the action part, or both). The task order was counterbalanced to avoid learning effects. In half of the composition tasks, participants had to check for recommendations. They were free to select the recommendation if it corresponded to what they had in mind otherwise, they could continue to create the automation as they liked. In the tasks that required participants to browse the recommendations, we counted the times that one was selected, as suggested in [21]. After the eight rule composition tasks, participants had to use the "Explore environment" functionality of the application, to inspect the automations they created and to assess which objects the rules were associated with. For this task, we counted a "success" if participants could correctly identify which automation was associated with an object and if any with multiple active automations was present. This task was performed using a separate application functionality ("Explore environment") not related to the presence of recommendations. In addition, participants had to respond to some statements on a 5-point Likert scale, where 1 corresponded to "strongly disagree", 3 "neither agree nor disagree", and 5 to "strongly agree" (see Table 2 for a summarization of the answers). The questionnaire yielded a Cronbach's Alpha score of 7.21, meaning that the responses are sufficiently reliable. Finally, they had to further elaborate by responding to four open questions. To better address the research questions, we formulated some hypotheses to be tested using the gathered data. The independent variables of the experiment are the versions of the application (with/without recommendations) and the different types of tasks participants had to do (simple and compound), while the dependent variables are the perception of ease, the satisfaction, and the time to complete the automation creation tasks. The hypotheses are: (Hyp1) There is no difference in the perceived difficulty between the creation of simple and compound automations. (Hyp2) Recommendations can make the application more satisfying to use. (Hyp3) Recommendations can make the generation of automations easier. (Hyp4) Recommendations can speed up the creation of automations.

5.2 Test results

The time participants took to complete the tasks was recorded considering each automation creation task individually (see Figure 5). The interval considered was from when they started to compose the rule with the application (after reading the task and thinking about what to do), to when they clicked the save icon at the task's end. The times show that in general tasks involving simple automations took less time than tasks with compound ones (Mean = 66 s, std. dev. = 27.60 s for simple, M = 105 s, std. dev = 36.55 s for compound), and using the RS increased the completion time for all the tasks (M = 78 s, std. dev. = 31.78 s without recommendations, M = 93 s, std. dev. = 41.69 s with recommendations), and also looking at simple (M = 60 s, std. dev. = 22.19 s without recommendations, M = 73 s, std. dev. = 31.04 s with recommendations) and compound tasks (M = 97 s, std. dev. = 28.84 s without recommendations, M = 113 s, std. dev. = 42.01 s with recommendations) individually.

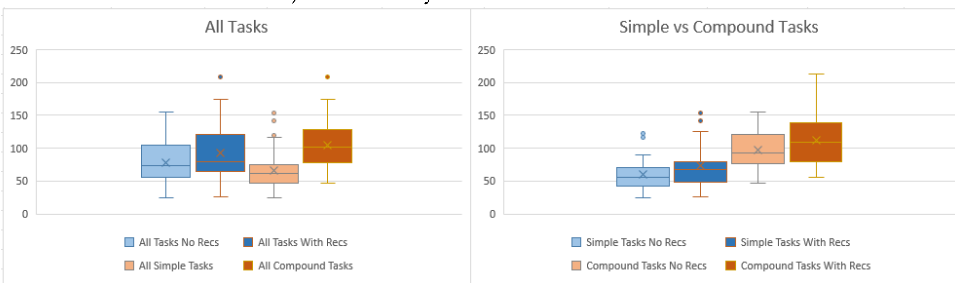

Fig. 5. Comparison of the task completion times using a whisker-and-bar plot.

Regarding Hyp4, we calculate the difference in these scores between tasks with and without recommendations. For the simple tasks, the normality of the paired differences between the two distributions was checked using the Shapiro-Wilk test, indicating no deviation from normality (W=0.980, p=0.834). Then, applying the Student's t-test resulted in a not statistically significant difference (t=-1.964, p=0.059). Concerning compound automations, using Shapiro-Wilk a departure from normality between the distributions was revealed (W=0.93, p=0.048). We hence checked for significance using the Wilcoxon signed-rank test, observing no significant difference (W=161.5, p=0.147).

Table 2. Results of the questionnaire with min, max, and median for each statement

Statem.	Text	Min	Max	Median
S1	Using AR to define automatons in a smart home setting is appropriate	2	5	4.5
S2	The approach used in the application to define automations is appropriate	2	5	4.5
S3	A smartphone is suited for this type of interactions	4	5	5
S4	Identifying which automations are associated with an object is easy	3	5	4.5
S5	Identifying which automations are associated with an object is a useful feature	4	5	4.5
S6	Using recommendations while carrying out the tasks is helpful	2	5	4
S7	Recommendations were varied	2	4	3
S8	Recommendations matched the automation's intended implementation	2	5	3
S9	The presentation of the recommendations was effective	2	5	4
S10	Configuring simple automations (1 trigger - 1 action) with the RS was easy	4	5	4
S11	Configuring simple automations (1 trigger - 1 action) without the RS was easy	3	5	5
S12	Configuring compound automations (more triggers and/or actions) with the RS was easy	3	5	5
S13	Configuring compound automations (more triggers and/or actions) without the RS was easy	3	5	5
S14	Using the application with the RS was satisfying	2	5	4
S15	Using the application without the RS was satisfying	4	5	4.5

Concerning Hyp1, we analysed the differences between the results of the perceived difficulty in composing simple and compound automations, to assess whether compound ones are perceived as harder. Regarding using the application without recommendations, no test was executed since participants provided identical scores to the perceived difficulty on the two tasks. For the version with recommendations, Wilcoxon signed-rank test was performed founding no statistical significance (W=2, p=0.773). Regarding Hyp2, we performed the same test on the satisfaction scores using the application with and without recommendations and found no statistical significance (W=27, p=0.222). For Hyp3, the same test was performed on the statements about the perceived difference in the ease of composing automations between with and without recommendations, with no statistical significance found (W=6, p=0.766 in simple automations, W=7.5, p=1 in compound ones).

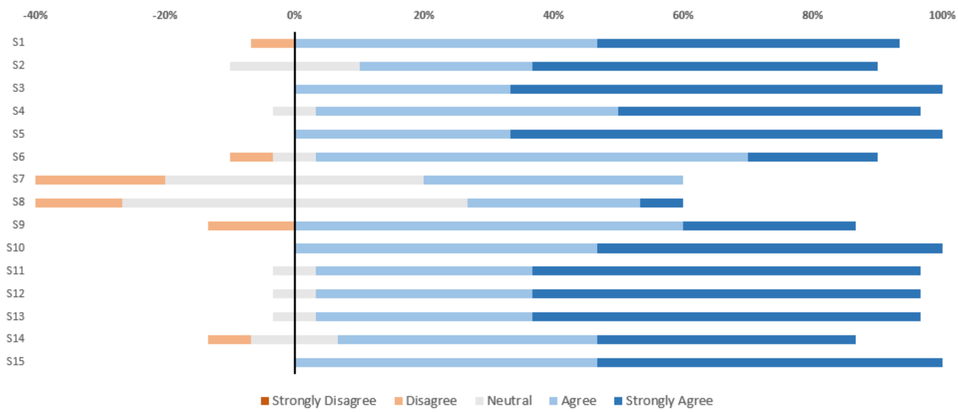

Fig. 6. Diverging stacked bar charts summarising the users' feedback.

Concerning the open questions, below is a summary of all the responses.

O1: Elaborate on the appropriateness of the proposed approach to generate automations, and how would you improve it. Ten participants found the approach completely appropriate, and have no suggestions for improving it. One participant suggested larger object markers so that they would be easier to select. Another proposed introducing some 3D visualisation for "simple" interactions such as light switches and augmented icons for objects with more capabilities. Other proposed improvements are using less schematic language and improving the intuitiveness of the UI. Finally, one participant argued that although AR is intriguing, a fully 2D interface would be more immediate to use.

O2: Elaborate on which device is suited for this type of interaction. All the participants agreed that the smartphone is the more suited device. Four participants reported that a tablet, having a larger screen, would also be suitable. Two participants mentioned the possibility of also using dedicated devices (HoloLens), but only if in the future they will become less uncomfortable and more affordable.

O3: Provide some feedback about the recommendations. Three participants cited ways to provide more personalised recommendations. These are: adding a panel with questions about user intent and modifying the recommendations accordingly, using the current or past state of the environment to refine them, and integrating data from Web services into the generation of recommendations, for instance, generating the text of reminders based on the user's calendar, Netflix preferences, or weather forecast. Two participants reported that the variety of recommendations is the main aspect to be improved. Other two proposed some "default" recommendations not generated from other users' behaviours. Participants were asked to elaborate on which aspect of the recommendations they liked the most and the least. The most cited pro was that they facilitate and speed up the composition (reported by six participants).

Another positive aspect is that they give new ideas about how to use objects and services (noted three times). The graphic of the augmented recommendations was also appreciated (two times). Finally, some users liked that recommendations were aligned with the automation they were creating (two times) and varied (once). As for the cons, the most reported issues are that they are not always relevant to the automation user is composing (four times), too few (four times), too few object recommendations compared to services recommendations (two times), and there is too little variety (two times).

O4: Elaborate on which aspect of the application you liked the most and the least. The most cited pro (by six participants) was the intuitiveness and ease of the approach. Another cited advantage (four participants) was the possibility to see all the created automations in the "Explore mode". Other cited positive aspects were the AR visualisations (in particular, the activation of the "exclamation marks"), the possibility to define complex conditions, the ability to discover objects' features and options through AR, and the graphics of the application. One participant noted that it was more satisfying to roam and set automations in the environment compared to sitting and using a desktop application. Concerning the cons, four participants noted that the 2D configuration panels could be improved. Three participants reported that some aspects of the application are not very convenient, namely, the need to tap the visualisations precisely, the need to switch between "create automation" and "explore environment" functionality, and that selecting more functionalities from the same object is not very intuitive. Also, there was some concern about the phone's heating up (noted by three participants).

6 DISCUSSION

In this section, we discuss our findings regarding the research questions identified in the introduction, the more general design implications that emerge from the experience, and the limitations of the study.

Concerning RQ1 (how to design an augmented-reality based EUD solution to allow users to easily create automations that can involve multiple triggers and actions), from statements S1- S3 and O1, O2 emerge that participants found the proposed solution appropriate to define automations. The mixed augmented objects / 2D panels approach and the procedure to define automations were found relevant, easy, and satisfying to use. Although the design of the proposed 2D interface is not in definitive form, participants overall had no difficulty in using the UI to configure the rule elements. Also, they did not perceive defining compound rules as more difficult than simple ones (Hyp1). For instance, one participant reported the possibility to compose complex triggers as her preferred feature. Several participants appreciated that there is no need to explicitly link the different trigger and action parts and that when a rule element has been configured you can continue by directly selecting the next one. This feature facilitates and makes more convenient the creation of automations. Separating the selection of the devices (in AR) from their configuration (with standard UI) could have contributed to this perception of easiness.

Regarding the RQ2 (how to exploit AR to identify the active automations and understand which devices they are associated with), from the results of task 9 and from statements 4-5 it emerges that the functionality that displays the descriptions of automations in panels placed over related objects allows users to easily understand the active automations and to which objects they are associated. All participants completed the related task (identify which automations are associated with an object) without any difficulty. Several participants reported this as their preferred application feature. It should be noted that this possibility is specific to AR and would not be possible with a traditional approach.

Referring to RQ3 (how to introduce the possibility to provide recommendations of possible automations in such AR platform), for the former, we observed that a paragraph embedding approach transforming text into vectors can be used to model TAP rules and generate recommendations. Transforming into text the various "data parts" of an automation leads to more precise recommendations with respect to a baseline greedy algorithm that only uses the class of the rule elements. About the latter aspect, from the responses to S9 and O3, we can observe that users generally appreciated the presentation of recommendations. According to Requirement 4, we observed that presenting recommendations in the AR space over the real corresponding objects can alleviate the problem of the scarcity of screen space, since they are integrated into the augmented real-world space. This feature was particularly appreciated by participants. Indeed, several of them reported that they would have preferred to see more recommendations in AR (for instance, viewing a list of recommendations for an object once they get close, including default configurations if there are no suggestions from other rules), and more balance between the ones that refer to objects and those to services (more common in the dataset, and hence more present in the recommendation lists).

About the RQ4 (evaluate the introduction of recommendation support in an AR tailoring tool), from the results of S6-8 and O3 it can be observed that in general participants perceived the recommendations as a useful feature, although there are some conflicting results in the user study. In the statements, the implemented RS received overall positive feedback and was perceived as useful to speed up and facilitate the process. Instead, it turns out that recommendations increase the time (Hyp4) to complete simple and compound tasks, although no statistical significance was found. However, there is no significant difference in the perception of ease of use and satisfaction between the two versions of the application (Hyp 2 and Hyp3). Also, some users perceived the recommendations as not in line with their customization objective, and not too varied. Nonetheless, in the tasks requiring browsing recommendations, these were selected 71.66% of the time. This indicates the effectiveness of providing recommendations in the rule composition process.

Some general implications can be drawn from the results of the test, the questionnaire, and from the feedback and observations of the participants. The first is that smartphone-enabled AR can be an effective approach to defining automations. Using a traditional tailoring platform, users have to learn an intermediate abstract mapping between the real objects they want to configure and their representation on the platform, e.g. through a hierarchical representation of the context or a long list of available functionalities. Making the association perceivable directly over the objects helps users identify the desired functionality, also making the process more enjoyable. Indeed, we observed that participants could easily select the intended objects and functionalities for instantiating the desired automation during the tasks. This result is consistent with previous studies where AR has been used to identify and select smart objects [9, 42]. Considering the growing design space of the possibilities of combining triggers and actions, providing users with clear ways to discover functionalities and communicate their personalization intents is crucial [12, 45]. Furthermore, AR can also make the process of connecting different objects more convenient, as also reported by [39]. All participants found the smartphone a viable device for the approach (indeed the statement "A smartphone is suited for this type of interaction" was highly rated). Overall, participants had no difficulties in creating simple and compound automations using the AR approach. It should be noted that previous AR approaches for configuring smart homes using TAP such as SAC [1] are limited to simple one trigger-one action automations.

Another finding is that augmenting the reality with abstract representations and information panels can be appropriate for the smart home context. Overall, we found that many of the features that participants appreciated most (the visualisations of the rule elements, the "recommendations panels", and the description of automations placed over the related objects) are specific to AR. For instance, participants reported that they would have preferred to see more objects recommendations (placed into the environment). This finding is in line with the work by Zimmermann and colleagues [53], in which AR recommendations were perceived as useful, entertaining, and informative. With respect to previous work, which shows the importance of the topic, the proposed solution in addition to end-user creation of automations also allows for the "exploration" of the environment, visualizing the automations currently associated with the objects. This functionality was particularly appreciated by participants and represents a step towards using AR to improve the transparency of home automation [48]. Another observation is that participants found relevant the abstract representations used to interact with objects. One participant mentioned the possibility of using more concrete visualisations (such as light switches or knobs) for some simple interactions, whereas another considered using pictorial representations of functionalities for a "beginner mode" of the application, but for typical use, they indicated the current visualisations as more suited. However, it should be considered that in order to draw more definitive conclusions, a comparative study of the different possible approaches (standard mobile application, mixing AR and mobile panels, application completely in AR) should be performed, also checking for discrepancies between the intended automation and the rules actually defined.

The last finding is that recommendations are perceived as useful during the composition process, but an approach solely relying on automations created by others may not be the most suitable solution. We observed a discrepancy between the perceptions that recommendations speed up the composition time, and the actual time recordings, which show instead a significant increase in time. However, this may be related to the experiment design, because participants have to go through all recommendations to check if one matches their configuration intent. Another aspect to note is that sometimes participants expected to find recommendations even if they did not insert any rule element or for objects with few stored recommendations. A possible solution is to generate some preferred defaults [22] to use when relevant recommendations are not available. Furthermore, some participants feel that recommendations would be more useful if more personalised. A way to solve this can be integrating data from more sources into the recommendation generation process, e.g., also using data from the environment. Deeper integration of recommendations with the context may be a relevant aspect of AR for smart environments. This integration of context data and AR using artificial intelligence has been studied, for instance, for personalising and improving the experience of visitors in museum curation [43]. Future development could consider this information to refine the generated recommendations, also considering further aspects such as personality traits, that in [8] have started to be explored concerning EUD environments.

Concerning the limitations, a study involving an installation in actual users' homes, for a prolonged period should be considered to obtain more complete feedback on the application, to investigate for instance how users reconfigure the automations in more realistic situations. Also, although recommendations were overall positively received, the number of automations rules from which they were generated was limited, hence possibly negatively influencing the perception of diversity and usefulness.

6 CONCLUSIONS AND FUTURE WORK

We have presented the design and implementation of an approach to defining automations in smart environments using AR. The solution also uses machine learning to generate rule elements recommendations related to the rule fragment configured by the user. The proposed solution exploits the ARFoundation framework to enable AR functionalities such as placing visualisations in the environment and interacting with them. Recommendations are modelled using the paragraph embedding approach through the Gensim library. The solution is an innovative way to support EUD in smart home systems, and received positive feedback in the user test carried out in one installation.

There are possible improvements that we plan to explore in future work. The main goal of this work was to assess the feasibility of the overall approach. As a continuation of this work, we plan a study with installations in actual users' homes during which we can also compare this solution with a traditional visual approach. We also would like to expand the used dataset, for example, using rules from other datasets, and to assess different text-based approaches for recommendations using other architectures (for instance, Transformers). In addition, participants found that recommendations would be more useful if integrated with more data, such as from the current and past state of the environment.

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

REFERENCES

[1] Ariano, Raffaele, Marco Manca, Fabio Paternò, and Carmen Santoro. 2022. Smartphone-based augmented reality for end-user creation of home automations. Behaviour & Information Technology, published online: 09 Jan 2022, pages 1-17, Taylor and Francis, https://doi.org/10.1080/0144929X.2021.2017482

[2] Narges Ashtari, Andrea Bunt, Joanna McGrenere, Michael Nebeling, and Parmit K. Chilana. 2020. Creating Augmented and Virtual Reality Applications: Current Practices, Challenges, and Opportunities. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20). Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376722

[3] Bak, Nayeon, Byeong-Mo Chang, and Kwanghoon Choi. "Smart Block: A visual block language and its programming environment for IoT." Journal of Computer Languages 60 (2020): 100999.

[4] Barkhuus, Louise, and Anind Dey. "Is context-aware computing taking control away from the user? Three levels of interactivity examined." In UbiComp 2003: Ubiquitous Computing: 5th International Conference, Seattle, WA, USA, October 12-15, 2003. Proceedings 5, pp. 149-156. Springer Berlin Heidelberg, 2003.

[5] Brackenbury, Will, Abhimanyu Deora, Jillian Ritchey, Jason Vallee, Weijia He, Guan Wang, Michael L. Littman, and Blase Ur. "How users interpret bugs in trigger-action programming." In Proceedings of the 2019 CHI conference on human factors in computing systems, pp. 1-12. 2019.

[6] 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.

[7] Castelli, Nico, Corinna Ogonowski, Timo Jakobi, Martin Stein, Gunnar Stevens, and Volker Wulf. "What happened in my home? an end-user development approach for smart home data visualization." In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, pp. 853-866. 2017.

[8] Cena, Federica, Cristina Gena, Claudio Mattutino, Michele Mioli, Barbara Treccani, Fabiana Vernero, and Massimo Zancanaro, Incorporating Personality Traits in User Modeling for EUD, Proceedings 3rd International Workshop on Empowering People in Dealing with Internet of Things Ecosystems, CEUR-WS.org, Vol. 3172, 2022.

[9] Meghan Clark, Mark W. Newman, and Prabal Dutta. 2022. ARticulate: One-Shot Interactions with Intelligent

Assistants in Unfamiliar Smart Spaces Using Augmented Reality. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 6, 1, Article 7, (March 2022), 24 pages. https://doi.org/10.1145/3517235

[10] 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 (September 2019)

[11] 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 Proceedings of the International Conference on Advanced Visual Interfaces, 1–9. https://doi.org/10.1145/3399715.3399905

[12] Corno, Fulvio, Luigi De Russis, and Alberto Monge Roffarello. "From users' intentions to IF-THEN rules in the Internet of Things." ACM Transactions on Information Systems (TOIS) 39, no. 4 (2021): 1-33.

[13] Joëlle Coutaz, James L. Crowley: A First-Person Experience with End-User Development for Smart Homes. IEEE Pervasive Comput. 15(2): 26-39 (2016)

[14] Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. "Indexing by latent semantic analysis." Journal of the American society for information science 41, no. 6 (1990): 391-407.

[15] Demeure, Alexandre, Sybille Caffiau, Elena Elias, and Camille Roux. "Building and using home automation systems: a field study." In International Symposium on End User Development, pp. 125-140. Springer, Cham, 2015.

[16] Desolda, Giuseppe, Carmelo Ardito, and Maristella Matera. "Empowering end users to customize their smart environments: model, composition paradigms, and domain-specific tools." ACM Transactions on Computer-Human Interaction (TOCHI) 24, no. 2 (2017): 1-52.

[17] Fogli, Daniela, Matteo Peroni, and Claudia Stefini. "Smart home control through unwitting trigger-action programming." In Proc. 22nd Conf. Distrib. Multimedia Syst.(DMS), pp. 194-201. 2016.

[18] Fogli, Daniela, Matteo Peroni, and Claudia Stefini. "ImAtHome: Making trigger-action programming easy and fun." Journal of Visual Languages & Computing 42 (2017): 60-75.

[19] 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.

[20] Ghiani, Giuseppe, Marco Manca, Fabio Paternò, and Carmen Santoro. "Personalization of context-dependent applications through trigger-action rules." ACM Transactions on Computer-Human Interaction (TOCHI) 24, no. 2 (2017): 1-33.

[21] Gunawardana, Asela, Guy Shani, and Sivan Yogev. "Evaluating recommender systems." In Recommender systems handbook, pp. 547-601. Springer, New York, NY, 2022.

[22] Haines, Will, Melinda Gervasio, Aaron Spaulding, and Bart Peintner. "Recommendations for end-user development." In Proceedings of the ACM RecSys 2010 Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces (UCERSTI), pp. 42-49. 2010.

[23] Valentin Heun, James Hobin, Pattie Maes. 2013. Reality Editor: Programming Smarter Objects. UbiComp (Adjunct Publication) 2013: 307-310J.

[24] Huang, Justin, and Maya Cakmak. "Supporting mental model accuracy in trigger-action programming." In Proceedings of the 2015 acm international joint conference on pervasive and ubiquitous computing, pp. 215-225. 2015.

[25] Dongsik Jo and Gerard Jounghyun Kim, ARIoT: Scalable Augmented Reality Framework for Interacting with Internet of Things Appliances Everywhere, IEEE Transactions on Consumer Electronics, Vol. 62, No. 3, August 2016.

[26] Sanghoon Kim and In-Young Ko. 2022. A Conversational Approach for Modifying Service Mashups in IoT Environments. In CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA. ACM, New York, NY, USA, 16 pages. https://doi.org/10. 1145/3491102.3517655

[27] Amy J. Ko, Brad A. Myers, Mary Beth Rosson, Gregg Rothermel, Mary Shaw, Susan Wiedenbeck, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scaffidi, Joseph Lawrance, and Henry Lieberman. 2011. The state of the art in end-user software engineering. ACM Computing Surveys 43, 3: 1–44. https://doi.org/10.1145/1922649.1922658

[28] Lau, Jey Han, and Timothy Baldwin. "An empirical evaluation of doc2vec with practical insights into document embedding generation." arXiv preprint arXiv:1607.05368 (2016).

[29] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Proceedings of the 31st International Conference on Machine Learning (ICML 2014), pages 1188–1196, Beijing, China.

[30] Toby Jia-Jun Li, Yuanchun Li, Fanglin Chen, and Brad A. Myers. Programming IoT Devices by Demonstration Using Mobile Apps. In Proceedings of the International, Symposium on End User Development (IS-EUD 2017). Lecture Notes in Computer Science, vol 10303. Springer, Cham. https://doi.org/10.1007/978-3-319-58735-6_1

[31] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf. 2006. End-User Development: An Emerging Paradigm. In End User Development, Henry Lieberman, Fabio Paternò and Volker Wulf (eds.). Springer

Netherlands, Dordrecht, 1–8. https://doi.org/10.1007/1-4020-5386-X_1

[32] Mäntyjärvi, Jani, Urpo Tuomela, Ilkka Känsälä, and Jonna Häkkilä. "Context studio: Tool for personalizing context-aware applications in mobile terminals." In Annual Conference of CHISIG, the Computer Human Interaction Special Interest Group of the Ergonomics Society of Australia, OZCHI 2003, pp. 64-73. University of Queensland Press, 2003.

[33] A. Mattioli, F. Paternò. 2020. A Visual Environment for End-User Creation of IoT Customization Rules with Recommendation Support. Proceedings AVI 2020 Conference, ACM Press

[34] A. Mattioli, F. Paternò. 2021. Recommendations for creating trigger-action rules in a block-based environment. Behaviour & Information Technology Journal, Taylor & Francis, Volume 40, 2021 - Issue 10, doi:10.1080/0144929X.2021.1900396

[35] Knierim, Pascal, Paweł W. Woźniak, Yomna Abdelrahman, and Albrecht Schmidt, Exploring the potential of augmented reality in domestic environments, in: Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI '19, Association for Computing Machinery, New York, NY, USA, Article 31, 1–12, 2019. doi:10.1145/3338286.3340142.

[36] Rehurek, Radim, and Petr Sojka. "Software framework for topic modelling with large corpora." In Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks. 2010.

[37] Salovaara, Antti, Andrea Bellucci, Andrea Vianello, and Giulio Jacucci. "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.

[38] Ronny Seiger, Maria Gohlke, Uwe Aßmann. 2019. Augmented Reality-Based Process Modelling for the Internet of Things with HoloFlows. BPMDS/EMMSAD@CAiSE 2019: 115-129

[39] Seiger Ronny; Kühn Romina; Korzetz Mandy & Aßmann Uwe (2021) HoloFlows: modelling of processes for the Internet of Things in mixed reality. Software and Systems Modeling, Available at: https://www.alexandria.unisg.ch/261944/

[40] Smyth, Barry, and Paul McClave. "Similarity vs. diversity." In International conference on case-based reasoning, pp. 347-361. Springer, Berlin, Heidelberg, 2001.

[41] Vijay Srinivasan, Christian Koehler, and Hongxia Jin. 2018. RuleSelector: Selecting Conditional Action Rules from User Behavior Patterns. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 2

[42] Evropi Stefanidi, Dimitrios Arampatzis, Asterios Leonidis, Maria Korozi, Margherita Antona, George Papagiannakis. 2020. MagiPlay: An Augmented Reality Serious Game Allowing Children to Program Intelligent Environments. In: Gavrilova M., Tan C., Chang J., Thalmann N. (eds) Transactions on Computational Science XXXVII. Lecture Notes in Computer Science, vol 12230. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-61983-4_9

[43] Torres-Ruiz, Miguel, Felix Mata, Roberto Zagal, Giovanni Guzmán, Rolando Quintero, and Marco Moreno-Ibarra. "A recommender system to generate museum itineraries applying augmented reality and social-sensor mining techniques." Virtual Reality 24 (2020): 175-189.

[44] 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.

[45] 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.

[46] White, Gary, Christian Cabrera, Andrei Palade, and Siobhán Clarke. "Augmented reality in IoT." In Service-Oriented Computing–ICSOC 2018 Workshops: ADMS, ASOCA, ISYyCC, CloTS, DDBS, and NLS4IoT, Hangzhou, China, November 12–15, 2018, Revised Selected Papers 16, pp. 149-160. Springer International Publishing, 2019.

[47] Wu, Q., Shen, B., Chen, Y. (2020). Learning to Recommend Trigger-Action Rules for End-User Development. In: Ben Sassi, S., Ducasse, S., Mili, H. (eds) Reuse in Emerging Software Engineering Practices. ICSR 2020. Lecture Notes in Computer Science(), vol 12541. Springer, Cham. https://doi.org/10.1007/978-3-030-64694-3_12

[48] Xu, Xuhai, Anna Yu, Tanya R. Jonker, Kashyap Todi, Feiyu Lu, Xun Qian, João Marcelo Evangelista Belo et al. "XAIR: A Framework of Explainable AI in Augmented Reality." In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, pp. 1-30. 2023.

[49] Yang R., Newman M. W. (2013) Learning from a learning thermostat: lessons for intelligent systems for the home. Proceedings of 2013 ACM conference on Pervasive and ubiquitous computing, 93–102.

[50] Imam Nur Bani Yusuf, Lingxiao Jiang, David Lo, Accurate Generation of Trigger-Action Programs with Domain-Adapted Sequence-to-Sequence Learning, In 30th International Conference on Program Comprehension (ICPC'22), May 16–17, 2022, Virtual Event, USA. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3524610.3527922

[51] Zancanaro, Massimo, Giuseppe Gallitto, Dina Yem, and Barbara Treccani. "Improving Mental Models in IoT End-

User Development." HUMAN-CENTRIC COMPUTING AND INFORMATION SCIENCES 12 (2022).

[52] L. Zhang, W. He, O. Morkved, V. Zhao, M. L. Littman, S. Lu, and B. Ur. 2020. Trace2TAP: Synthesizing Trigger-Action Programs from Traces of Behavior. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 4, 3, Article 104 (September 2020), 26 pages. https://doi.org/10.1145/3411838

[53] Zimmermann, Robert, Daniel Mora, Douglas Cirqueira, Markus Helfert, Marija Bezbradica, Dirk Werth, Wolfgang Jonas Weitzl, René Riedl, and Andreas Auinger. "Enhancing brick-and-mortar store shopping experience with an augmented reality shopping assistant application using personalized recommendations and explainable artificial intelligence." Journal of Research in Interactive Marketing 17, no. 2 (2023): 273-298.

Received: January 2023; Revised May 2023; Accepted: June 2023.