POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Conversational Agents for Creating Personalization Rules in the IoT

Supervisors

Luigi DE RUSSIS

Alberto MONGE ROFFARELLO

Candidate

Carlo BORSARELLI

Academic Year 2020-2021

Acknowledgements

I just want to thank all the people who have accompanied me in these 5 years.
First, I thank my supervisors Luigi De Russis and Alberto Monge Roffarello who supported me in this thesis. I did not know how to write formal and technical documents in English; they guided me in the writing and in the organization. It was very formative. Their professionalism, promptness, and patience allowed me to carry out and complete a project that I didn't really think was possible.
I thank all my fellow students, in particular Eleonora, Stefano, and Paolo. Great companions, but above all friends, they made the days shorter and helped me when I was in trouble. Edoardo, Cesare, Francesco, Tarik, Giacomo, Andrea, and Francesco, dear friends. They are so different from each other, but all of them are a precious part of who I am. Thanks for giving me unforgettable memories.
These past two years have been difficult for me. Loneliness, work, and doubts have weighed me down like I didn't think were possible. I thought myself stronger, but I was not. For this reason, I want to thank my family for the extreme trust placed in me, they never doubted that I could do it. They supported me a lot and motivated me to conclude this journey with no kind of oppression or limitation, thanks to Mauro, Rossana, and Elena mean a lot to me. I wanted to dedicate the last few lines to Clara, who made every moment precious together, who has lightened all the load I had, who every time makes me smile and feel good like the first day, thanks for the unreachable support.
I'm afraid, but I can't wait to know what's waiting for me.
Thanks.

Abstract

Smart speakers, such as Google Home or Amazon Echo, are entering our homes and enriching the ecosystem of the Internet of Things (IoT) already present in them. The Intelligent Personal Assistants (IPAs) they include allow users to ask for different information (e.g. the weather or a recipe), set reminders and lists, and directly control other IoT devices (e.g. lamps). These assistants, through a companion app installed on the owner's smartphone, provide advanced features like the possibility to set up some personalization rules in the form of trigger-action: if something happens, then do something else. In alternatives, there are visual programming tools such as IFFTT or Node-RED that allow for more complex management of IoT by creating also complicated rules, but these tools lack simple management and require some effort by less technical users. The voice processing capabilities of the IPAs and their knowledge of the IoT ecosystem in which they are inserted could be exploited for the creation of the rules through a purely vocal interaction.

This thesis aims to explore novel approaches for creating personalization rules through conversation between the user and an IPA. To this end, interviews were held focused on identifying a set of strategies used to create rules in trigger-action format by voice. Later the thesis illustrates the design and the implementation of two prototypes with different rules composition approaches: one completely based on voice interaction and the other requiring physical action on home devices to show the IPA what to do. The work continues with the evaluation conducted through an in-the-lab experiment with 10 users; the evaluation focused on the usability and the efficiency of the conversational agents, then we have carried out a comparison between the interfaces based on the considerations and the results achieved by testing.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The number of smart speakers on the market, and their diffusion in homes, is growing more and more. The demand for home entertainment is increasing even during the historical period of the COVID-19 pandemic. The latest estimate, conducted by Canalys, highlighted that the global smart speaker market will reach 163 million units during 2021, with a growth of 21% compared to 2020 [1]. The cost of these devices has now become accessible to everyone, and the race for the best smart speaker is still underway — the most bought speakers are Apple HomePod, Amazon Echo, Google Home. A recent study conducted in 2021 [2] identified the main reasons that led so many customers to have a voice assistant in their homes. In this context, a paper has developed a technology acceptance model for smart speakers [3]. The analysis focused on the perceived usefulness, its enjoyment, the risk of the technology (anxiety to be recorded) and the ease of use. The results showed that among the variables, enjoyment had the strongest effect on the behavioral intention of using smart speakers, thus leading to mass diffusion.

1.1 The Advent of Virtual Assistants in the IoT

Barricelli in her study [4] conducted a discussion regarding the evolution and the advent that has led to the spread of increasingly integrated intelligent environments, with particular regard to new approaches to end-user development activities. The paper identified how voice assistants are gradually replacing the traditional GUIs. The assistants are in fact able to solve some tasks that are typical of the most common human skills; as by reading texts or emails, scheduling appointments, setting reminders, or even giving little tips of shortcuts to recurring tasks. Barricelli detailed also that [4] it is clear how the importance of virtual assistant has grown leading to a change of definition: from Virtual Voice Assistant to Virtual Personal Assistant (VPA) or Intelligent Personal Assistant (IPA), the latter not only perform the tasks mentioned above but are able to collect data of the user and store them in the cloud, learning from his preferences, until they become communication channels between the user and the Internet of Things (IoT) ecosystem. IoT is generally defined as a network of connected devices that offer IT services, by offering many possibilities for interaction between the environment and users. Also defined as smart things, the IoTs entities constitute support to the execution of the operations of daily life [5]. This combination of technologies, including IPAs, can be applied to smart homes where the user lives and could benefit more from the support in daily activities.

1.1.1 Smart Home Configuration

The Intelligent Personal Assistants present on the market as Amazon Alexa [6], Google Assistant [7], and Siri [8] bring users closer to their smart homes and allow them to communicate with certain IoT devices. Turning on the light, lower the shutters, or set a temperature are all examples of actions that a smart home already allows by a conversation with an IPA. But the possibility to set a more complex scenario in a smart home underscores the necessity to introduce the concept of End-User-Development (EUD). The EUD was defined by Karat and John:

> *"a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify, or extend a software artifact" [9]*

In this context, this thesis discuss EUD with respect to the IoT domain, EUD can offer to end users tools that allow them to configure some complex behavior of their smart home [10].

Fogli et al., in the article [11], review the literature of smart home tools, they show that the trigger-action rules represent the most used paradigm for smart houses personalization. Also, according to Yu et al. [12] the Trigger Action Programming (TAP), necessary for trigger-action rules creation, is one of the powerful End-user development frameworks that simplify IoT automation. I have to mentioned some tools as IFTTT [13], Zapier [14], Microsoft Automate [15], Apple HomeKit [16], Node-Red [17]. Such software allows users to perform trigger-action development by defining rule in the construct of "if-then": *if* something happen *then* an operation is executed. Also, an assistant like Amazon Alexa provides the possibility to create some personalization rule through a companion app installed on the owner's smartphone, but this feature remains hidden for most people because requires some time and patience to navigate in the setting and configure all rule details.

All the TAP systems mentioned are based on a visual interface with easier or more difficult rule creation systems, but none of those systems exploits the potential of the voice interaction that we think can integrate ease of use and flexibility in a smart home context. In this thesis, I want to expand the IoT ecosystem customization trying to find different approaches for the personalization of the IoTs, all by focusing on the vocal. The idea is to provide tools to the inhabitants of the houses that allow them to adapt their behaviors according to their needs by exploiting the potential of the vocal approach.

1.2 Project Goals

More specifically, this thesis explores novel approaches for creating personalization rules through a conversation between IPA and user. In particular, the goals are to identify a) if a no-technical user would be willing to create a rule just by speaking with a smart speaker, b) what are natural formalisms to compose rule by voice, and if the trigger-action paradigm is good or not, c) which features should have IPAs to be feasible for the creation process.

1.3 Report Structure

- Chapter 2 focuses on the state of the art in the project's field. The reviewing of the works in the research area will help to identify hints on the development and the implementation of the thesis work.
- Chapter 3 describes the conduction of the semi-structured interviews with 7 non-technical users. First, I took place background interviews, then with an imagination exercise, I explored how users would create personalization rules by talking with a smart speaker.
- Chapter 4 analyzes the high-level prototyping. Starting from the knowledge gained with the interview and the research phase, I designed 2 solutions with 2 different rule composition approaches, one completely vocal, and the other that requires the user action on the home devices to show the system what to do.
- Chapter 5 describes the technologies and the implementation techniques used to develop the conversation agents.
- Chapter 6 analyzes the evaluation process of the prototypes in order to validate the usability of the interfaces. I conducted 10 usability tests with non-programming users; it was used a within-subject design solution, which allowed participants to test both prototypes.
- Chapter 7 focuses on the final consideration of the project by analyzing the major difficulties, the successes of the project, and exploring the future work.

Chapter 2

Background and Related Works

This chapter aims to expand the context of this project by reviewing works in our research area. In order to identify how IPAs can support users in the personalization of their IoT ecosystem, I start with Section 2.1 by reporting the advantages of Natural-language programming and why it is important to invest in it. Later, the following Section 2.2 specifies the integration of smart speakers in some End-User Development contexts by exploring some projects. Finally, Section 2.3 describes some recent works that explore the creation of trigger-action rules in smart home contexts with some chat-based platforms.

2.1 Natural Language Programming

Nowadays there are several architectures and many languages which allow the technical user to program, but more and more, we live in a digital world that needs to be customized and adapted to end-users' needs, also for those without programming knowledge. Natural Language Programming wants to make machines significantly easier to use. Back in 1978, Dijkstra claimed that if we design machines that could be instructed in our native tongues, life will become easier [18]. Also Gordon et al. [19] said that the natural language interface used as an intuitive programming language may have a major role to play in the future. If we think of the documentation in software engineering of today's applications, in a way, in Natural Language Programming this becomes the final program. Programming is still too exclusive. We all use applications every day, but few people designed them. The idea is to expand the possibility of programming with natural language that we can hypothesize that as technology, it will play a larger role in people's lives.

I mention the work of Brummelen et al. [20]; they designed a conversational agent that allows the user to develop software programs just by conversing in natural language with a voice-based system. Brummelen proposes a solution with a multi-turn program synthesis and addresses questions about the usability and cognitive load of a conversational programming system. To achieve this, the prototype (named Convo) was evaluated with a user study that highlighted its criticalities and problems of use. It turned out that the speech synthesis system is still a problem today. Even if you use the best one (in this case was used Google Cloud Speech-to-Text), there are still problems in speech recognition. Brummelen pointed out that ambiguity reduction techniques are needed, such as conversational QA or immediate feedback from the agent. The user test noticed how advanced participants (experience in programming) strongly prefer the text-based system over the voice-based system. Also, many critical issues related to the lack of visual feedback are mainly due to the cognitive load required. The results show that Convo is pleasant to use for inexperienced people and the system may be a better fit for an educational introductory tool rather than an advanced tool.

It can be extrapolated that, increasing accessibility and reducing the cognitive load can allow as many people as possible to access programming with natural language, especially non-programming users.

2.2 Integrating Smart Speakers in EUD

Current IPAs have the possibility of being integrated into platforms capable of performing some complex features, as the creation of trigger-action rules, without the need to programming. In addition, there are several powerful frameworks in the market that gives the possibility to connect smart devices, APIs, and online services. Solutions as IFTTT [21] or Zapier [14] are some of the trigger-action frameworks, they easily allow the creation of automated tasks across different platforms. In addition, they integrate IPA in the creation of rules (*"Recipes"* in IFTTT and *"Zaps"* in Zapier) which are mainly used as triggers.

Rahmati et al. [22] and Corno et al. [23] have compared the most popular trigger-action programming frameworks in their studies, they have found that these are a versatile and powerful tool but limits the integration of smart-speakers as a trigger of previously programmed rules. The article [24] combines an Alexa personal assistant by integrating it into a framework for creating / editing / deleting rules. However, here too, the prototype named TAREME (Trigger-Action Rule Editing, Monitoring) mainly uses the IPA as a trigger or as the target of an action (e.g., playback of voice messages). Recently, Rajalakshmi et al. [25] and Kodali et al. [26] propose 2 home automation solutions in which IPAs are integrated into platforms as IFTTT or Node-Red [17], the systems designed allow to connects and controls most IoT devices using voice. The elasticity of the systems and the positive results of the projects allow us to understand how the personalization of the home environment combines well with the voice interaction. This thesis tries to expand this context by using IPAs not just as actuators of personalization rules, but also in the creation process.

2.3 Voice-based Trigger Action Programming

De Russis et al. [27] exposed that the possibility of a conversational approach in the management of an IoT ecosystem, as a smart home, has led to many developments in the field of research. A first example is InstructableCrowd [28], a crowd-powered system that allows users to create if-then rules through conversation with crowd workers. The backend is managed by crowd workers who have the task of creating rules by asking follow-up questions to the user. A second project called HeyTAP [29] is a conversation-based programming platform capable of mapping user needs by extrapolating If-then rules from them. The user can provide HeyTAP with intentions that are used to recommend IF-THEN rules, afterward, the user will choose from the proposed ones. The above mentioned platforms allow the definition of rules but the interactions are in any case strongly based on screen and the user could not create rules directly, but only by confirming the solutions proposed.

The trigger-action model has attracted a lot of research with many applications and systems, as ParlAmI [30] a multimodal conversational agent that is able to create rules using natural language and a GUI. The project underlines how the interaction is simplified by Conversational Interfaces, and the users can interact with spoken language in a natural way. ParlAmI is based on a framework named LECTOR an advanced tool that identifies user behaviors and supports them with the creation of trigger-action rules. ParlAmI is based on a chat-bot that tries to extrapolate from user sentences the information needed to LECTOR to create a rule. A user study results in positive user opinion about the system, underlining how the conversational approach with rule creation is a good alternative and particularly easy to use.

A recent work [31] develop a conversational interface to manage an IoT ecosystem. The prototype, named Jarvis, offers a very advanced chatbot that supports actions with a lot of devices, from simple ones (e.g. Turn on the kitchen light) to more advances trigger-action rules (e.g. If it is 7 am turns on the light), integrating all of them in a more advanced system. This work proceeded with an evaluation experiment which resulted that Jarvis is preferable to low-code solutions.

2.4 Summary

The development of home automation technology has led to many in-depth and extensive studies on home configuration systems. As widely identified by the study [32], the solutions currently on the market are characterized by high complexity, and often require the intervention of a technical user or computer expert. This excludes many family members in the process, making complicated the configuration of the house for less technical consumers or simply less passionate about technology. Reflections and works presented in this chapter have confirmed that conversation can be a great way to bring the user closer to programming, and is also perfectly applicable to the EUD in the IoT. In addition, if the interaction were simpler, more people would approach the customization of their smart homes. This thesis job wants to focus more on trigger action rules, which I believe (also with the support of reported works) is the easiest way to personalize an IoT ecosystem.

Since the reviewed works remain strictly connected with a display-based system as a chatbot or a complete GUI, I focused this project on IPAs integrated into smart speakers, also to follow their extremes spread in people houses. Although conversational agents such as Siri, Alexa, and Google assistant are already present in dedicated applications or integrated directly into the operating systems (macOS / iOS / Android), this work wants to focus on standalone smart speakers. Studies show how smart speakers are the gateway to the smart home [33], so I decided to focus on this approach to personalization especially since this thesis work focuses on conversational interaction directly with home devices. The following chapter presents the semi-structured interviews conducted in order to gather information on how users create personalization rules vocally. The obtained results have contributed to the prototyping and the designed phases of the two rule composition approaches.

Chapter 3

User Study

This chapter presents a user study in the form of an interview with 7 non-technical users. It is reported the design of the study in which it was asked participants to conversate with a smart speaker to personalize their hypothetical IoT ecosystem at home. The interview also analyzed the attitudes of the participants towards home automation. This was useful to understand if there are differences in approach between those who use and appreciate home automation and those who do not. Then, a discussion of the results is presented in order to analyze how the information acquired could be considered in the design phase.

3.1 Method

For the purpose of drawing as much information as possible from the research, I structured the following study using a semi-structured interview. Semi-structured interviews allow interviewees the freedom to express their views on their own terms. I interviewed one participant at a time with open or closed questions accompanied by follow-up questions. In particular, I have taken a free approach to the questions trying to get as much information as possible from a relatively unexplored in research field. In this way, participants had more freedom to share their thoughts, thus giving us the opportunity to investigate hypothetical situations that the participants simply did not experience.

Since the semi-structured interviews contain open questions, reasoning, and thoughts, the interviews were video recorded, and subsequently, I transcribed them for analysis. The recordings have started after the consent of the participants. Recordings are also necessary because taking notes focusing on the participant's responses was difficult, so I proceeded in this way to better conduct the interviews and lead the discussion based on the responses.

The interviews were conducted in Italian, both via Zoom video call [34] and in person, they took place in the north of Italy in April 2021 with 7 users; interviews lasted from 25 to 40 min.

3.2 Recruitment

I have recruited potential participants through social channels such as Facebook and Instagram; the choice was concentrated on potential participants from my social circles, both because of the pandemic situation and for the possibility of having more confidentiality and naturalness in the answers. In order to have a broader spectrum of technological affinities and background studies, I carried out the interviews by targeting a heterogeneous group of interviewees. An online questionnaire guided the selection of participants, it was created and issued through Google Form [35] with the aims to identify their sociodemographic characteristics. We were interested in:

- **Age**: age is generally negatively correlated with the acceptance of technologies. During the survey, the participants were provided with 7 age groups to choose from.

- **Job position/study background**: the wanted people were those who did not have a background or job positions exposed to advanced technology, the research of this study aimed to identify mainly non-technical users.

- **Level of satisfaction/home automation interest**: in order to understand if the participants were already aware of this technology and the general level of interest, I restricted the search for participants to those who had some interest in home automation. The indication was placed with a 5-point Likert scale.

- **Smart speakers' owner**: the interviews were conducted by taking a quite equal number of participants who own and do not own smart speakers. In addition, I asked a question to identify the frequency of use of a voice assistant; the question was useful to understand how confident the participants were to talk to a conversation assistant.

- **Level of self-evaluation technological skills**: users provided on a 5-point scale, the level that they considered corresponding to their technological personal skills. I used questions like "*Can I successfully search the Internet for information?*" and "*I am comfortable with the use of streaming TV services (such as Netflix, Prime Video, Disney +)*".

3.3 Participants

Seven participants were selected for the interview; I present their characteristics and resulting demographics in the Table 3.1. Their ages ranged from 18 to 52, all the participants were Italian, 4 women, and 3 males.

Participants and characteristics					
Participant	Age Range	Job	Smart speaker's owner	Home automation interest	Technological skills (avg)
P1	23-27	Speech therapist	✓	4	4.75
P2	18-22	Student	✓	5	4.0
P3	48-52	Housewife	✓	5	4.5
P4	23-27	Health technician	✗	4	4.75
P5	23-27	Student	✗	4	3.75
P6	43-47	Health workers	✗	3	3
P7	23-27	Math teacher	✓	4	4.5

Table 3.1: Demographic information and Smart home familiarity of interviews participants

The analysis of the questionnaire allowed the identification of the background and demographic information of the participants. Among the 7 identified, I found no one to be an IT expert; the school training was different, mostly scientific, and by chance many users worked in the health field. The average technological self-evaluated scores make understand that respondents had a certain technological awareness and familiarity with IT devices. The choice also focused on medium-high interest in home automation, in order to be sure to have a significant impact on the involvement of the interviewees in the study. I attached the interviews questions and the followed script in the appendix section of the thesis (Appendix A).

3.4 Interview Results

The interview was divided into 2 main sections; the first investigated the interviewees' relationships with smart speakers, if they had them, what they were using them for, and what features they knew. Interviewees were asked to respond to their experience with home automation systems, the question asked if they own smart equipment in their homes or if they ever used a smart device.

After the background interview, I conducted an imagination exercise where participants were put in a position where they needed to talk with a hypothetical IPA in order to create a personalization rule. The information collected will allowed us to explore the possibilities and approaches that a non-technical user would use to create personalization rules through conversation.

3.4.1 Familiarity with IoT Ecosystem

At the beginning of the interview, the participants were encouraged to tell about all their experiences with smart speakers. The owners of smart speakers (P1, P2, P3, and P7) all had an Echo Dot model, so they were used to conversing and interacting with the Alexa Intelligent Personal Assistant. Smart speakers have been used in different contexts. P2 and P1 described a primary use for listening to music and for entertainment in general; while P3 used this device to avoid loneliness:

> *"It's fun, it keeps me company. I search for stuff on the web. I play music, add items to the shopping list ..."*

P7 mainly used it to set alarms and timers:

> *"I often use Alexa to set timers, it's probably the most comfortable and effective method I know of, both cooking and setting alarms before falling asleep."*

P1 described an interesting experience of smart speakers:

> *"My patient, with severe motor disabilities, uses the device a lot, in particular, who cares for him uses it to involve the patient, this is since he cannot read and use mobile devices. He could have a way to control alarms or reminders. The speaker involves him as an active part of the day and makes him interact with the something."*

P2 and P7 were the only participants who currently use smart speakers connected to smart devices, they listed smart bulbs and led strips. All the users also used the IPAs integrated into smart speakers for web searches.

Even the participants who did not have a smart speaker had already seen one or tried to interact with it. The best known and most used is the Amazon Echo Dot, P4 said he used it a few times at the friend's home.

I then asked participants the smart speaker's functionalities that they knew, all of them listed similar operations. The most mentioned were the interaction with the smart home environment, the online searches, the calculations, the timer and alarms setting, and listening to music. That confirms that all participants had a basic knowledge of the technology so the imagination exercise would have been more effective.

After that, I asked participants their experiences with smart devices, if they had ever interacted with ones or not. P3 and P2 have a smart home, they said they had quite the same devices such as shutters, anti-theft, door, gate, garage doors, lights, radio, room broadcasting. The 2 interviewees live in a home automation environment with the possibility of implementing scenarios and automation, but they are not the family members who directly interact with the gateway in order to configure scenarios. They never programmed their smart home because they find the interaction complex. Their interactions are limited to simple commands like turn on the lamp or close the shutters. P3 said:

> *"I only learned to turn the alarm on and off when I enter the house, I set the temperature once. Nothing else."*

P2 knew the possibility of connecting the 2 smart bulbs he has with his smart house but he never tried because he had to install a separate software:

> *"The electrician told me it was possible, he also tried to explain to me but it seemed complex, I gave up..."*

P1 told his thoughts based on the experience she had at a friend's house:

> *"I think they can be very comfortable. They shorten the time and also the fatigue. But if you put a smart speaker in people's hands who are unfamiliar with them, paradoxically, they lengthen the times instead of shortening; it is necessary to learn how to use them by living with them and getting used to the comforts they can give."*

To introduce the participants to the second section of the interview, I made them imagine plausible scenarios in a smart house. In order to become familiar with the concept of rules, I asked them which processes they would prefer to automate in a hypothetical home automation house. The responses vary from the automation of the heating system when returning home (P5, P3) to the shutters and alarm system automatically active when all family members were at home. Participants were not found particularly interesting to automate the lighting systems, only P1 mentioned that he would like to have automatic lights at the entrance to the house. The action of turning on lights is not considered so exhausting that needs for automation.

It was asked if participants knew the advanced features of modern smart speakers to set personalization rules through the proprietary app. Only P7 and P1 and P3 responded positively, P1 only knew of their existence but she had never tried to create any. P3 thought of interesting rules but replied that he didn't get to set one. Only P7 actually created a rule:

> *"I tried to create one that recommended the app, saying "Alexa, goodnight" all connected lights should turn off and some relaxing music should started ... setting this rule was simple because the app guided me well, but I've never wanted to create others, I didn't want to spend so much time managing on the app."*

After these questions, I explained the purpose of the thesis and the interview research to them in detail. Each participant responded positively to the idea of configuring the rules directly through their smart speakers, they thought it can solve many problems that previously prevented them from configuring scenarios. There were no particular differences in reception between those who owned and did not own smart speakers.

3.4.2 Creating Rules Through Conversation

After the background questions, I asked participants to perform an imagination exercise in which they were prompted to imagine to conversate with a smart speaker to personalize their smart homes by creating personalization rules. I asked them to imagine a hypothetical scenario of a fully smart house, with potentially one speaker per room and sensors for the passage between the rooms with home automation devices.

First Rule Formulation Attempt

The scenario posed to the participants was:

> *"You would like to turn on the central kitchen light every time you enter that room"*

The rule that the participants tried initially to create had to be pronounced from the smart speaker in the kitchen. Then I asked them if they would have wanted to reformulated the rule and how, for example, if they had to create it from another room such as the bedroom.

All participants started the sentence with "*Alexa*", the command necessary to ensure that an intelligent speaker that integrates Alexa could respond; this reflects how this command is so widely spread in people's minds.

I found a difference between the answers from those who own a smart speaker and those who have not. In particular, the answers given by P1, P2, P3, P7 (smart speaker owners) can be summarized as:

> *"Alexa, Every time I walk into the kitchen, turn on the central light."*

These participants still made the specification of the room, even if I specified that the rule was created in the kitchen and that the device was in the same room. The previous experiences of these interviewees caused this formulation. And, as a consequence, they said they wouldn't change the rule if they should talk to the speaker in the bedroom (always referring to the rule in the kitchen). They thought the rule contained everything needed to create the behavior in any room.

The other participants gave different answers. For example, P3 tried to say:

> *"Alexa, turn on the light every time I pass by."*

P6 replied similarly:

> *"Alexa, whenever you see someone walk through the door, turn on the main light."*

As P5:

> *"Alexa, when you feel that the passage is activated turn on the center light."*

None of these mentioned the kitchen, and P3 totally emitted the specification of the type of light to turn on. The answers given above had produced, as expected, a more precise reformulation of the rules if these would be created in different rooms. The formulations to the same rule created from the bedroom were like the previous ones of the owners of smart speakers.

Second Rule Formulation Attempt

The scenario posed to the participants was:

> *"Do you want to set the fact that when you go to bed the shutters of the house are closed and all the lights go out"*

Compared to the first scenario participants reacted by thinking more time about the rule formulation. They were unsure how to make clear when they would go to bed, they asked me questions like *"Do I have to set an hour?", "What if one day I go to a different time?","Are there any sensors on the pillow?"*

P3 tried to say:

> *Alexa, every evening after 10 pm when I go to my room, pull down the blinds and turn off all the lights.*

So, the participant wanted that the rule was triggered only after a certain hour. However, she said that because of her habit of entering the room only for sleeping. But she had added that her daughter at 20:00 is already in the room, so the same rule could not fit for her.

P5 tried to imagine a sensor on the pillow and replied:

> *"Alexa, when you hear me go to sleep, pulls all the shutters down and turns off all the lights."*

But she portrayed, she asked yourself that if one day she went to bed in the afternoon, how would she do it? She wouldn't have wanted it all shut down.

P1 came up with the idea of a command:

> *"I would like him to close and turn everything off but I would have to tell him something, like a command, I could attach the command to a word like: Alexa, every time I say goodnight turn off all the lights of the house and lower all the shutters."*

The choice of command (e.g. "*goodnight*" or "*I'm going to sleep*") was partly guessed and partly suggested to the other participants. All of them, after the hint, preferred the command to any other type of automation with the second rule.

P4 made this observation:

> *"If I don't have the same routine, I don't care that something happens at the same time every day. If one works shifts and has only one sleep window like the one in the afternoon every other day. Perhaps with the word/command, it is more versatile in terms of use."*

However, P6 raised a concern about the use of commands:

> *"It would probably be difficult for me with too many commands, I don't think I could remember more than 3/4."*

The Speakers' Answers

I proposed to all participants some potential speaker answers to the first scenario of the rule in the kitchen. I listed to interviewees three main speakers' responses, in which the IPA had different levels of automation and awareness of the context.

The presented answers were:

1. The speaker automatically sets the rule without asking you for confirmation (understanding automatically the room and the devices)

2. The speaker reformulates the rule according to its grammar and repeats it in order to understand if it has acquired all necessary specifications. For example:

   > *"The rule just set refers to the KITCHEN room, it will turn on MAIN KITCHEN LIGHT, if the KITCHEN MOTION SENSOR is activated, correct?"*
   >
   > *[YES / NO]*

3. The speaker breaks down the message into individual blocks, allowing the user to confirm or withdraw each. And after repeats the entire created rule as in the second answer. A possible situation:

   "Does the rule just set refer to the "KITCHEN" room?"
   [YES / NO]

   "The light you want to turn on is MAIN KITCHEN LIGHT?"
   [YES / NO]

   "The passage sensor is KITCHEN MOTION SENSOR ?"
   [YES / NO]

No user commented positively on the first IPAs' response. Although in the previous phase, before presenting the aforementioned assistant answers, I asked participants what they expected the IPA to answer, no one tried to imagine complex answers as the ones presented above.

P1, also given to his profession, commented on the first proposed answer:

> *"I think it is not functional. It does not give you the opportunity to verify that what you want has actually been understood, based on voice recognition it may happen that it does not understand. That it also makes me think of people who do not articulate well, it is something that must be taken into consideration!"*

The other participants reasoned in a similar way, no one saw this scenario as possible. They could be afraid of having spoken badly, so they wanted to be sure of the words said. For this reason, all the interviewees preferred the other 2 answers.

In particular P1, P4, P5, P6, and P7 had preferred the second answer, they had the possibility to double-check what was sensed by the assistant and eventually confirm it. They had listed several reasons for this choice. In the first place, it was appreciated the repetition of the rule. Second, the third was seen as a boring answer, they appreciated the compromise between speed and reliability proposed by the second one.

P2 told me how the third answer gave him more security. The participant said that the creation of a rule is important for life in a smart house, and is not too bad to wait for more. Because of the answer given by P3, who in the conversation emitted to say the type of light; so, before asking for the speaker's answer, I asked him what he would have expected from the speaker to face off with device specification:

- With a certain level of automation, the speaker tries to automatically identify the device (the bulb) and respond directly as in the scenario above.
- The speaker asks the user for the light to be turned on, without any kind of advice.
- The speaker asks the user what light it was by listing those in the room.
- The speaker asks the user to manually turn on the desired light, in this way the speaker would understand directly which device he intended.

P3 preferred automation, saying:

> *"Better if the speaker try to guess the light, where is automated. Instead, I would prefer that it breaks the answer as in the third above scenario, better if it specifies them well so if I did not understand, it's more clear. In case I don't compose the rule right I realize it immediately, in the second I would have to repeat everything and for long commands, I might not remember all the details. So I prefer the third, in such a way I'm more sure it understands."*

So the participant preferred a certain level of automation, without light specification, and that the speaker would respond as in the third response scenario.

Wrong Device Recognition

I proposed to the participants a situation in which they realized a mistake made in the composition. So when they should have replied "NO" to the speaker's rule confirmation. Here, the case of the wrong light was proposed, in which the user was wrong to specify the smart light in the creation of the rule.

I then presented answers such as the previous ones listed for P3. This makes clearer how participants would face off an error in composition. The most chosen method is trial and error in which the participants rephrased the sentence, or a portion of it (as the third case). P1 does not see the possibility in which the speaker lists all the lights in the room, too long and frustrating. The participant added that, in general, people act actively and would rather have repeated the rule 10 times. Even P2 and P3 thought that eventually, they make themselves understood by IPA if they try and try, also P6 and P7 would do the same.

The manual turn on and off of the desired device from the specific speaker request was seen as intricate to imagine and not very intuitive, although it could probably be the fastest.

I then presented the case of many devices in the room, most of the interviewees preferred that the speaker would try the "auto-completion" attempt. If instead, the devices of the same type were few or single, the participants would prefer that the speaker asked for device specifications with, eventually, options to choose from.

3.5 Discussion

The goal of this interview was to explore how non-technical users would create custom rules via conversation with an IPA of a speaker.

The interviewees first described their general experience with smart speakers by indicating their knowledge and familiarity with the IoT ecosystem. After some initial questions to make the participants understand the study context, I presented the focus of the interview in detail and I proposed scenarios where users had to try to interact with the IPA.

The first result is that the participants do not consider the possibility of a fully automated system. In discussing with the interviewees, they all found a minimum dialogue necessary, both because they admitted they could make mistakes by speaking, and because not thought they knew exactly what they wanted from the rule. On the other hand, it was interesting to see how the interviewees, however, also reacted negatively to the possibility of a "non-automated" speaker. The idea of took a long conversation with a speaker, in which someone would have to specify every detail step by step, was seen as improbable.

The solution to this problem of creating rules is therefore an automation compromise, the spectrum of the responses of the participants was wide, the interviewees gave more or less automation capacity to the speaker.

From responses collected in the question on expected speakers' answers, there was no clear correlation between who had chosen the second response scenario and who had chosen the third one. P2 and P3, who had preferred the third scenario, both have a smart speaker, but P7 and P1, who don't have, had chosen the second.

Neither age nor work background had a relevant effect on responses.

According to P1, P6, and P7 the person wants to act quickly anyway, they wouldn't wait for the device long talking. According to them too many details asked and a too-long conversation with the device could be tedious, they gave an indicative limit of 2 or 3 questions maximum. Another problem encountered, particularly by P5, is that a system that takes a long time to set up a rule could demotivate the user to create other rules, he would easily lose the desire to interact.

P2 and P3 instead had a different idea, they thought that since setting a rule is one-off, it wouldn't have been bad to have a little more conversation. For them, it was good that each portion of the rule was specified, in such a way the user checks each detail; the system would have been more effective and solid.

By analyzing the second rule proposed, participants who did not think they had a fixed routine had preferred to associate the rules with a command. However, as analyzed by P6, it would not make sense to have more than 3/4 commands linked to scenarios/rules. It is important to find a compromise between the important commands that can be related to the routine and their number.

On the other hand, it is interesting how the users, who do not own a smart speaker, were more natural not to specify the room of the rule. Instead, the opposite happened for users with speakers, which underlines their habit of specifying details of the sentences: they were used to expect some misunderstanding. P1 said that it is the person that adapts to a smart speaker, not vice versa. A list of solutions to solve a speaker's error is unthinkable, a user would instinctively try again until the speaker understands him.

On the other hand, P3 said he would also accept a list of elements by the assistant but not over 2/3, otherwise, the situation would be frustrating.

All participants created rules using a similar trigger-action structure, thus confirming how this formalism is good and versatile.

In the following chapter, I used the information collected by expanding the user study considering more approaches for the rule composition. Before starting the implementation I designed some high-level prototypes in form of conversational flows, by exploring two different rule composition approach.

Chapter 4

High-level Prototyping

A conversational approach has many possibilities and potentials, but this does not suit every type of task. The initial research phase and people interviews have shown how a conversational approach for the personalization of a home environment could be intuitive and effective for most people.

Considering the timing and development effort, I have reduced the use cases to support. I focused on a single smart house scenario and on the most common use cases. This chapter starts by describing the design phase of the conversational interfaces. I chose to prototype two different conversational agents, each corresponding to a different rule creation approach. During the high-level design of a speech-based application, I followed and analyzed the main conversational components. In this way, to make conversational interaction more reliable, I covered the principal components identified by the literature of conversational interfaces. For the in-depth study and design guide of the main conversational components I used: "*Conversation Design*" by Google [36], "*Voice Design Best Practices*" by Amazon [37] and the book "*Designing Voice User Interfaces*" by Cathy Pearl [38].

4.1 Solutions Overview

The user study has shown that creating trigger-action rules by voice is practicable and natural, especially for non-programmer users. Also, the ones less experienced in the interaction with IPAs found easy the vocal composition of the personalization rules, thus confirming the applicability of natural language in EUD in the IoT. The interviews revealed that users prefer a system with an automation compromise, so the IPA should be exhaustive in rule presentation and in the specification of rule parameters, but it should not exaggerate with the requests, confirmations, or with unnecessary details. In the interviews, participants did not show particular preference on the IPA responses, for some users act quickly was fundamental, the rule creations divided into multiples steps would result in a too-long conversation for them. Other participants have reacted positively to a longer conversation, they said that specify all the rule sections in detail was not a problem for a one-off action. For these reasons and to test and compare the two systems, the design considers two different approaches, one that considers the rule creation in one step, the other that requires more steps to compose the rule. For both prototypes, I chose a level of automation that can help the user, but without specifying unnecessary information. The sections below show all the details: from the choice of which confirmations are needed, to the specification of the rule missing parts.

In interviews, the physical action on the IoT device, in order to tell IPA what the device to add in the rule, was not seen as a possible choice. The fact that the interviewees didn't think of a solution as the physical acting on the device, can be correlated to the inexperience in doing that kind of action, in today's applications is not common to interact on a physical device to set automatic behaviors, especially in a smart home context. Instead, the voice interaction is considered a normal operation, so it is not a strange thing to do for the users. I hypnotized that a solution that can explore also this novel way of interaction (the physical action on home devices) and this different approach to smart home personalization, can be really interesting and useful. In particular, I chose to design one prototype completely vocal, in which the rule creation consists of one step. The second prototype requires user physical action on the IoT device to ensure that IPA detects univocally the home device and its state, this agent has a rule creation divided into two steps, first the composition of the trigger then the action. The thesis proposes two prototypes of conversational agents that use different approaches to the creation of personalization rules. They will first be designed, then implemented, and evaluated.

This chapter starts presenting the designs in the form of conversational flows and then it analyzes the various components of different conversations which lead to rules creation.

4.2 Prototype 1

I designed Prototype 1 by considering a full-vocal interaction: the IPA (the assistant inside a smart speaker) will ask the user to compose an entire rule vocally. In this rules creation approach, the user should pronounce both the two components of the rule, the trigger and the action in a single sentence. In natural language is always better to consider all situations: if some rule part is missing, if the user requires help or if a user wants to modify the rule before setting it. For example, it can happen that the user forgets to say the action or specify the device type.

In case of a rule missing part, the IPA will reply asking for the specification of rule details. A possible sample dialog in which the user forgets to specify the light device:

- User: *"If I enter the kitchen, turn on the light."*
- Assistant: *"Which light do you want to turn on: sink light, main light, or all the lights?"*
- User: *"The second."*

Conversations can take a huge number of paths, each person can answer differently to the same question. The desire is that someone that uses the interface should be trained less as possible; the prototype should map what users can say in any situation, this is done by defining as many conversational flows as possible.

I start by presenting the conversation flow diagram of the first voice interface (Figure 4.1), it provides a general structure of the possible conversations between the user and the IPA. Then I proceed with a components analysis of each conversational step.

4.2.1 Welcome

In voice applications, the user must be informed by the assistant of the specific actions that could perform. In this case, it is necessary to present the instructions on how to create a rule. In the beginning, at the first start, the IPA will provide information, and summarize the possibilities of the interface. Once the user has become familiar with the commands and logic of the voice interface, it is no longer necessary to provide detailed instructions on what the application can and cannot do. A user can be considered an expert user, if already has created multiple rules, so the IPA doesn't need to present him again how to create one. Furthermore, it is necessary not just to count the number of uses, because, after months, the user may have forgotten the instructions provided the first time. The IPA, after a certain period, should repeat again the rule creation process.

Voice interaction leads the user to be forced to listen to all that is said, while in visual applications, the user can select the parts of his interest and go straight to the point. In the design of a voice application, it should be studied well how to present the information to the user in the most complete way but without being verbose, the user could not skip anything.

I provide an example of how Prototype 1 could provide the initial instructions in order to allow to introduce new users to application grammar and logic:

- User: *"Hey Google, I want to create a rule."*

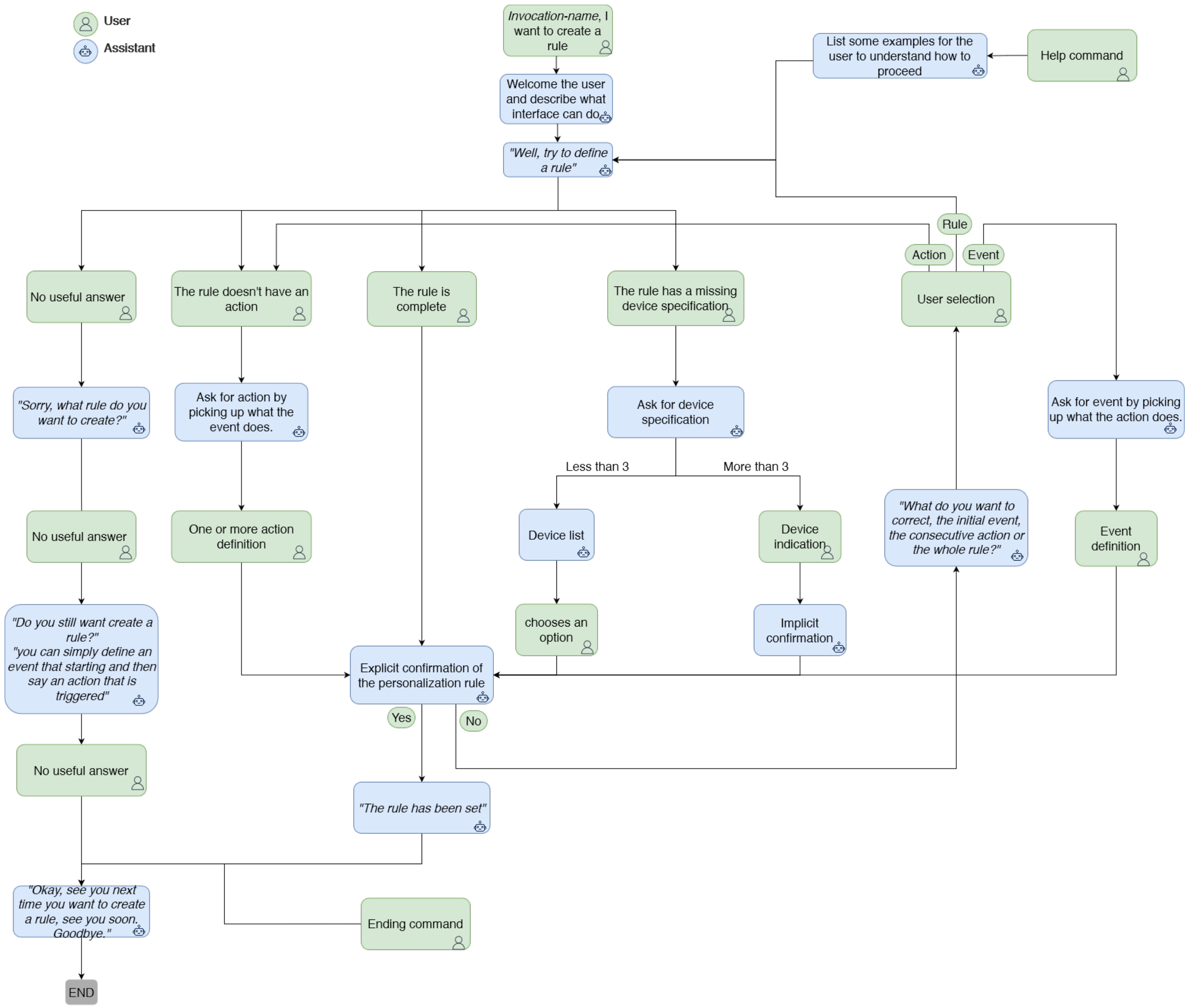


Figure 4.1: Conversation flow of the prototype 1

- Assistant: *"I will help you personalize your home automation system by creating rules. To specify a rule, you need an event that triggers a corresponding action. You can say, for example: when it's 10 pm, close the front door, or, When I* *say goodnight, turn off all the lights in the room. Now try to define a rule."*

In the above example, I hypnotize a situation of a new user that wants to create a rule for the first time by talking with a speaker with Google Assistant integrated.

4.2.2 Help

The Help component is used to guide users in cases where they are in the creation phase of the rule and they do not know or do not remember what to do. By saying *"help"*, the prototype will provide all instructions needed to proceed, for example by providing some example sentences of rule creation:

- Assistant: *"Welcome back, try to define a rule."*
- User: *". . . what should I say?"*
- Assistant: *"Well, to specify a rule, you need an event that triggers a corresponding action. You can say, for example: if it's 11 pm, close the shutters, or, every time I enter the kitchen, turn on the central light. Now try to define a rule."*

In this case, the help provided by the IPA is similar to the introduction of new users (as the example in Subsection 4.2.1). A user who does not use the application for a long time could forget the basic operations, the IPA should act in order to support him.

It is therefore useful to adapt to the context of the conversation at that precise point in which help is requested. Users can say the help command in different ways, the prototype will not force them to say the exact word *"help"*, but just saying phrases like: *"what should I do?"* or *"how do you create a rule?"*, the help section is triggered. A possible use is when, in the creation phase of the rule, an user does not know or does not remember what to do.

4.2.3 No Useful Input

In the design of a conversational interface, we should be careful about the errors that could occur in the conversations. The happy path imagined (ideal scenario without error conditions or exceptions), is not the only way people could use the prototype to create a rule.

First, I want to analyze the absence of words error. When the user does not know what to say, is distracted, or speaks too softly, the IPA does not detect any useful voice input from the user. The prototype requires an explicit response in order to proceed, creating the rule requires steps that must be performed in sequence, so the user's response is needed. For this reason, when the user does not provide useful input within a predetermined time, the IPA will call the user explicitly asking to reply, and possibly repeat the interaction. In case the user does not provide a useful answer after 3 attempts, the application will ends. At first attempt, a simple ask for repetition can be enough, but if the user for two times doesn't provide anything that can be interpretable as a personalization rule, the IPA will increment the help by providing some example and guiding lit of bit more the user. A similar case could happen when speech recognition (ASR - Automatic Speech Recognition) returns incorrect results. Here the user responds but the IPA cannot recognize anything relevant, it will be handled as in the previous cases, asking the user to repeat.

4.2.4 Missing Rule Details

As also emerged from the interviews and the research phase, one of the major problems in these types of applications based on voice interaction is the lack of clarity with which people naturally converse. Even in everyday life, when we talk to someone, we omit a lot of details that often require follow-up questions to understand what we intended.

There may be cases where users do not specify a device that is present in multiple instances within the room, as in the common case of the light. For example, if a user during the creation of a rule says: *"at 11 pm turn on the light of the kitchen"*, it is a possible rule, but if kitchen lights are more than one, what should the IPA choose?

The case above is solved in Prototype 1 by requiring light specification. As also introduced in the sample dialog above (Section 4.2), the agent will list the IoT devices of the type indicated by the user, like:

- User: *"At 11 pm turn on the light of the bedroom."*
- Assistant: *"Which light do you want to turn on: the bedside light, the central light, the cabinet light, or all the lights?"*

If there are over 3 devices, IPA will ask a general question e.g. *"which light do you want to turn on"*. This is because listing 10 different lights do not lead to a quick and pleasant conversation. In this case, the system could be more prone to errors: if the user does not know the device name with which the IPA saved it, problems could arise.

When devices are unique in the smart house, such as a smart coffee machine or a smart lock-door, even if the user does not provide the room or device specification, the IPA will not require explicit clarification; the missing information can be derived from the spacial context (more details in Subsection 4.2.7). Non-essential questions only add friction to the user experience. For this reason, no further questions or even implicit confirmation phrases are needed.

If the user forgets to say the room of the rule and this cannot be deduced from the context, the IPA will ask for the room specification:

- User: *"If it's morning, turn on the light."*
- Assistant: *" In which room do you want to create the rule?"*

In the example above, the lights in a smart house are hardly ever unique devices, so IPA should ask the user to specify the room, also why listing all the smart house lights is not feasible.

The problem does not arise when the user wants to select all the devices of a single type in the house like: *"if it's morning, pull up all the shutters"*.

4.2.5 Confirmations

To design a good voice user interface, the correct method should be found for confirming the user input, by finding the right balance and not using excessive confirmations which could frustrate people. A confirmation ensures the user what they have just said and, if needed, allows to correct any error committed. As reported in the book [38] the correct confirmation strategy requires the analysis of several factors:

- **Consequence of the error**: when a rule is set differently than the user's wish, it could negatively affect life in a smart home. It is, therefore, important to ask for explicit confirmation before setting the rule.
- **Methods for providing feedback**: confirmations are a way to provide feedback. In this thesis, I focused on smart speaker devices which are audio-only devices. In order to avoid increasing the user's cognitive load, the IPA must provide them feedback. But it is important to not exaggerate by overwhelming people with too much information and answers.
- **Appropriate confirmation type**: there are different confirmation, the explicit one that forces the user to confirm the information to proceed, and the implicit one that let the user know what the IPA has understood, without explicitly asking for confirmation. In Prototype 1 the best choice is the explicit confirmation because of the importance of the rule in a smart home. The IPA before setting the rule will ask users to confirm what they pronounced.

Prototype 1 requires much information in a single sentence: a trigger, a room, an action that comprises a device, and its state. Providing confirmation can tell users what they just said, allowing them to understand if each rule parameter is as they have imagined it. I presented a possible sample dialog:

- Assistant: *"Hi, try to define a rule."*
- User: *"Close the shutters in the living room every day at 10pm."*
- Assistant: *"You have defined that each day at 10 pm, the living room shutters will close. Do you want to confirm the rule?"*
- User: *"Yes."*
- Assistant: *"The rule has been set, goodbye."*

The IPA will reformulate the rule basing its grammar, so the user can understand well if the rule that has just been pronounced can be implemented. A "yes" from the user, allow the IPA to set the rule in the smart home ecosystem. Instead, a "no" requires the activation of a rule correction procedure that I expose in the next section.

4.2.6 Corrections

The user should have the possibility to correct the entire rule. The correction procedure starts when the user is misinterpreted by the IPA or, simply, if the user mistakes in rule creations. There may be multiple parameters to edit, so one-step correction is not preferable. The assistant may have misinterpreted both the event and the actions of the rule, I chose to make IPA ask what users want to correct and allow them to change just the trigger, the action, or the entire rule. I show an example of a use case where a user wants to correct the rule before setting it:

- Assistant: *"Okay, if someone enters the kitchen, the central light will turn on, Is it correct?"*
- User: *"No"*
- Assistant: *"What do you want to correct, the initial event, the action, or the entire rule?"*
- User: *"The action"*
- Assistant: *"Alright, what do you want to happen each time someone enters the kitchen?"*
- User: *"That the sink light turns on"*

- Assistant: *"Great. So the rule is: if someone enters the kitchen, the sink light will turn on, confirm the rule?"*

The correction procedure doesn't force the user to restart the entire dialogue from the beginning. Repeating parameters already correct is particularly frustrating and time-consuming. As in the example above the user does not need to repeat the trigger. Users should correct just what they want. Also, repeating complicated rules from scratch with many actions can be difficult.

4.2.7 Context

A fluid and functional conversation requires a certain IPA's awareness of the context. The literature of conversation agents considers two main contexts, the spatial context, and the conversational context. A good agent should have both. The spatial context allows the IPA to know the room where it is placed and to interpret a user sentence like: *"turn on the light of this room"*, the assistant will understand that *"this"* means the room where it is. Instead, the conversational context is relative to the entire conversation between the user and the IPA. The assistant must be able to adapt the subsequent questions to the previous answers, basing the conversation on what was previously said by the user.

An example of conversational context, is when a user wants to correct the action of the rule, the IPA should help the correction as in the example above *"alright, what do you to happen each time someone enters the kitchen?"*, here the IPA takes back the already created event and asks the user to correct the action. This allows making the conversation with the smart speaker more natural.

4.2.8 A Way Out

In Prototype 1, I have added an ending command. A user can quit the conversation for any kind of reason just by saying *"goodbye"* or *"exit"*, *"cancel"*,*"stop"*, after that the voice application will close, even in the middle of the interaction. When a user exits the conversation before confirming the rule, the eventual rule portion already created will be deleted.

4.3 Prototype 2

Prototype 2 is slightly different from the previous one, and it exploits the fact that the smart speakers have the knowledge of all the IoT devices in the smart home where they are in. For example in the case of lights, the IPA already present on the market can answer the question *"is the kitchen light on?"*. In order to take advantage of this possibility and to explore a different approach for the rule creation process, I designed Prototype 2.

In this conversational agent, the rule creation is split into two, the user to complete the rule formulation has to start with the creation of trigger and just after the action is defined. The trigger is created entirely by voice, instead, the action requires the user physical action on the IoT device that the user wants to add in the rule. I show in the Figure 4.2 the conversation flow diagram of Prototype 2, which describes the high-level interactions between users and the IPA. Many sections of this prototype are similar to the previous ones (like the ones in Subsection 4.2.2, Subsection 4.2.3, Subsection 4.2.7 and Subsection 4.2.8), since I used the same approach to create the vocally part of the agents, I report here only the main differences.

4.3.1 Welcome

As in Prototype 1, the user is initially guided to the creation of the rule. But, the instructions and the examples explain just how to create just the event. Also in this conversational agent, the IPA doesn't present again the instruction to the expert user.

4.3.2 Action Creation

Prototype 2 requires a separate analysis for the creation of the actions. As already said, the action is created from the user's physical interaction with the real smart device. After the user setting of the trigger, the IPA will require the creation of the action by asking for the user's physical interaction with the device. The user will set the IoT device he wants to add to the rule to a desired precise state. I provided only a few seconds for the user to act on the device, after which the IPA will ask again if the user is willing to act or not. In case the user confirms the request, other seconds are given, otherwise the conversation ends. In particular, I chose that about 10 seconds is a good time for the deadline, more details are in Section 5.5. As soon as the user interacts with a device the IPA will detect the one chosen and the state set. I show a sample dialog to describe the action creation:

- Assistant *"So, do you want to confirm that the rule triggers every day at 8 pm?"*
- User: *"Yes."*
- Assistant: *"Well, you have set the trigger, now act directly on the device you want to add in the rule."*
- User: (pull down the bedroom shutters).

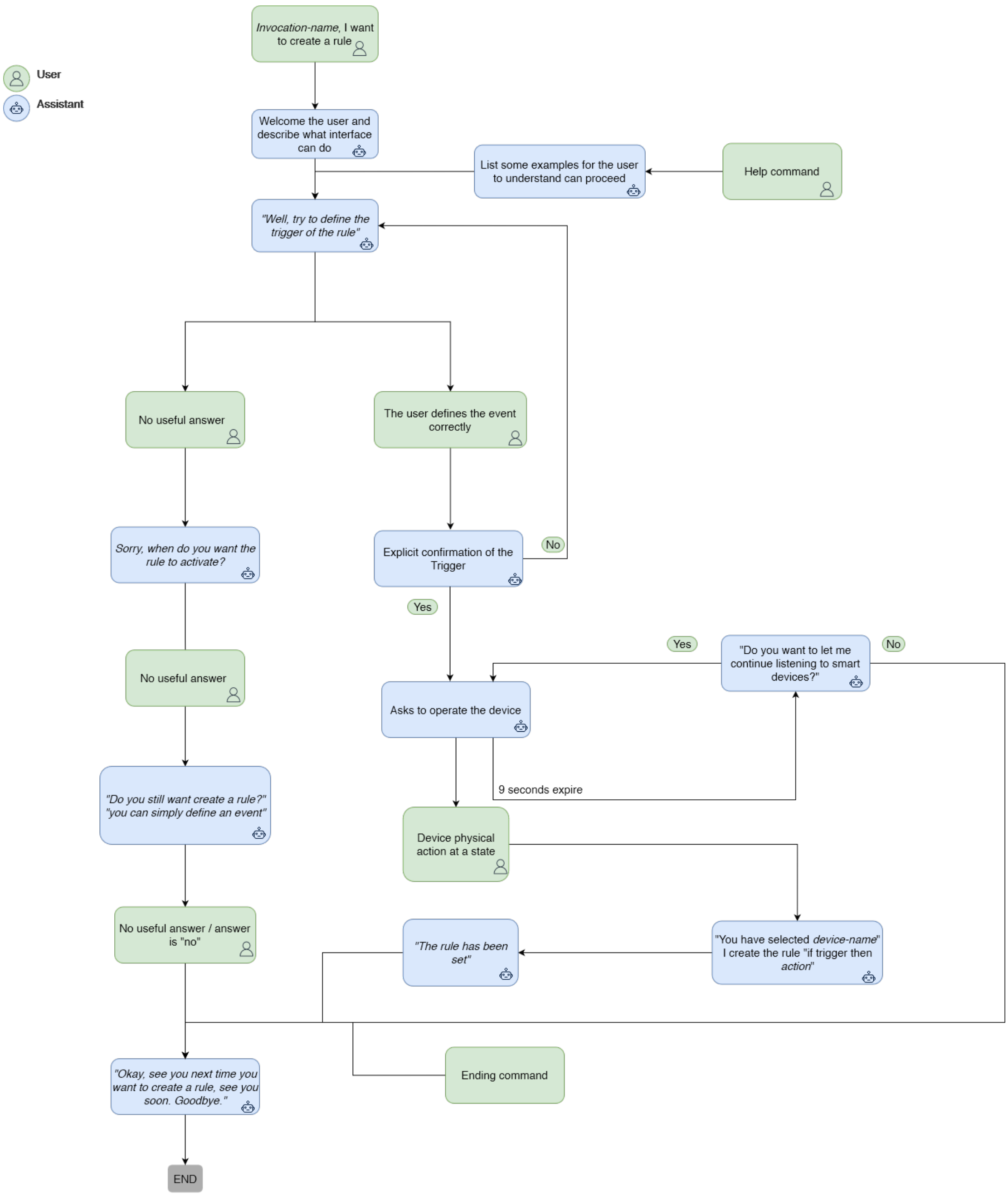


Figure 4.2: Conversation flow of the prototype 2

- Assistant: *"Got it, you close the shutters in the bedroom."*
- Assistant: *"Now, every day at 8 pm, the bedroom shutters will close."*

The example focuses on action creation, I present a sample dialog for trigger creation in the next section.

4.3.3 Confirmations

Since the user interaction with an IoT is a simple action, I choose to not insert an explicit confirmation of the action part of the rule. However, explicit confirmation is required for trigger creation, while implicit confirmation is used for the action definition. So, after the user interaction with the device, IPA will say the device name and its state, in order to implicitly confirming the user action. As mentioned above (Subsection 4.2.5), a confirmation ensures the user what they have just said. I presented a possible sample dialog, that covers both the confirmations used:

- Assistant: *"Hi, when do you want to activate the rule?"*
- User: *"Each time someone enters the kitchen."*
- Assistant: *"So, do you want to confirm that the rule triggers every time the KITCHEN MOTION SENSOR is activated?"*
- User: *"Yes."*
- Assistant: *"Well, act directly on the device that you selected."*
- User: (turn on the light that he wants e.g. sink light).
- Assistant: *"Got it, you turn on the sink light."*
- Assistant: *"Now, each time someone enters the kitchen the sink light will turn on."*

The above example shows how final explicit confirmation is not required, when the user confirms the trigger and acts on the device, a mistake is almost impossible. Instead, the user that doesn't want to set the rule can simply terminate the conversation before physical acting on the device.

4.3.4 Corrections

In Prototype 2 the correction is simpler than in Prototype 1, I choose to allow the user only to correct the trigger of the rule, so the correction system will not require any kind of choice between parameters. The action is not easily corrected, because of the almost impossible misinterpretation by the IPA: if the user turns on the light manually, is not possible for the assistant to interpret another device.

4.4 Conclusions

The designed interfaces allow providing more advanced ways for any kind of user to create personalized scenarios in their home. Considering a novel approach for rule creation, I have designed and implemented 2 conversational agents that help to this goal. The thesis has chosen two types of rule composition, the first fully vocal takes the advantage of natural languages by developing a natural conversation for any kind of smart home inhabitants. Prototype 2 requires an active part of the user who must interact with the smart house to make the assistant aware of his intents. By exploiting the IPA possibility to knowing all device's states, the second conversational agent does not require the formulation of the entire rule in one step. In Prototype 2 the user should just say little commands, so I hypothesized that this rule creation approach can be less fast than the one in Prototype 1, but probably less error-prone. The conversational design can make IoT devices management an easy and accessible choice, especially for non-technical users. The next chapter will present the implementation of the conversational interfaces, by discussing all the technologies used in the development of two real conversational agents that will be tested and evaluated in a smart home scenario.

Chapter 5

Implementation

This chapter describes the implementation of the two prototypes designed. While the high-level prototyping chapter focused on the conversational components and on the high-level interaction, the implementation chapter goes through all the processes needed for the voice interfaces to actually work with the end-user. The chapter starts with an overview of the technologies and platforms used, then each component is described from the backend tools to the user interfaces.

5.1 Architectures Overview

The last link in the chain is represented by the Action for Google Assistant, which composes the main technology used for managing the user input and output phrases. Natural languages processing is handled by Dialogflow, the platform is useful to capture the user intents in order to find matches with preloaded information. The most important intents, cached by Dialogflow, are sent to a webhook service with the intent management. The Nodejs webhook is the core of both conversational agents, it is responsible for the management of the user expressions, and, basing on those it prepares an adequate response. In Prototype 2, Nodejs webhook is also in charge of detecting the user's physical interaction on the smart device. I used the Firebase platform for different purposes, by linking to the same account of Dialogflow agent, I exploited multiple tools like:

- **Cloud Functions for Firebase**: serverless framework used to connect the agent to the backend webhook, it is responsible for the HTTPS requests management.
- **Real-Time Database**: a NoSQL database used to archive and synchronize all the information relating to the IoT devices present in the simulated home. The data is stored by the system in JSON format.

- **Firebase Hosting** (only Prototype 2): hosts the resources of the web application developed to simulate the IoT devices.

The Figure 5.1 and Figure 5.2 describe the process flows of the two agents created by including all the main technologies used. Further information is reserved for the following sections.

Figure 5.1: First high-level architecture of the full-vocal prototype

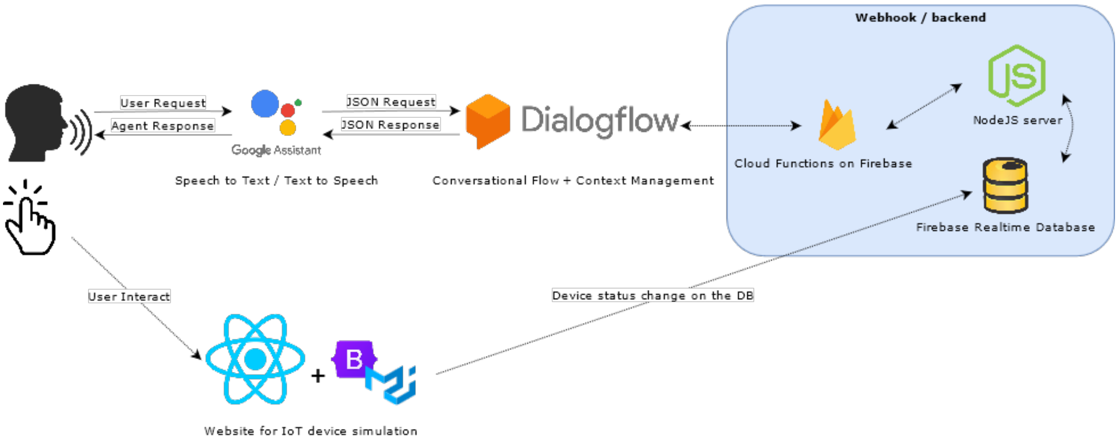


Figure 5.2: Second high-level architecture of the prototype that require user interaction on IoT devices

5.2 Dialogflow

In the implementation, I chose to use Google Cloud services, the account and the subsequent creation of a Google Cloud project allowed the integration of many powerful tools, such as the use of Dialogflow and the simple combination with the Firebase platform.

5.2.1 Agent Creation

The start of a conversational agent takes place with the creation of a new agent in the Dialogflow console. This platform allows managing both the processing of natural language (sent as input by the user) and the automatic learning. Prototypes development focuseed on creating voice interfaces so, once I connected the Google account with the Dialogflow console, I created both the agents with this platform.

The input is handled via JSON files, they are sent to the Nodejs backend webhook for the processing of response, when a response is created, the corresponding JSON file is sent back to the agent. After the received message from the backend, Dialogflow will send to the input and output device (the smartphone or the smart speakers with the IPA) the instructions on what to say, then the conversation moves on to the next turn.

In the agent languages section it can be added more languages that the Dialogflow voice interface can understand. I have chosen to set Italian as the only language. The intents, described below (Subsection 5.2.2), have to be trained with multiple phrases that should be created in consultation with a native speaker. It is not enough to translate the phrases into the language destination, but it is necessary to adapt to the context and the culture of the country destination of the prototype. It is therefore important to focus on a single language. I also created both the agents and the test in Italian for the possibility of evaluation on Italian soil and to facilitate the recruitment phase.

5.2.2 Intents

Each turn of conversation with the user is marked by one intent, in order to manage a complete conversation each agent comprises multiple intents. Whenever the user says something, Dialogflow associates the expression with one and only one intent, which is considered the best one for the agent. For each turn of the conversation and for each prototype I have defined multiple intents, everyone has the following components:

- **Context**: in order to handle an expression, can be provided an input and/or output context to correctly match an intent. In this way a developer can control the flow of a conversation. When an intent is found from the user's expression, all output contexts are activated. When contexts are active, Dialogflow finds intent configured with the input contexts corresponding to those currently active. To better direct the flows of conversations, I have used a lot of input and output contexts. In this way the user, even by mistake, cannot say an expression corresponding to the intent of too successive (or too previous) conversational turns. That has helped to follow the directions defined in the high-level conversations designed in the previous chapter.

For example, the input context of the intent *"P1 - Get Rule"* (from Prototype 1) is *"awating_rule"* and the output context is *"explicit-confirmation"*, the latter will instead be the input context of intent *"P1 - Explicit rule confirmation - no"* and *"P1 - Explicit rule confirmation - yes"*. This allowed agents to remember the parameters' values and pass them to subsequent intents. And also it forces the conversational flow to only 2 intents.

- **Training phrases**: the intent is chosen by the agent if the user's expression is similar to any of its training phrases. The training phrases should, as much as possible, cover all the approaches to with users could answer the questions. In the development phase, I paid greater attention to the intents *"P1 - Get Rule"* (Prototype 1) and *"P2 - Get Event"* (Prototype 2) which represent the intent of the creation of the rule of the first prototype and the creation of the event of the second one. The interviews and early testers helped to expand the training phrases.

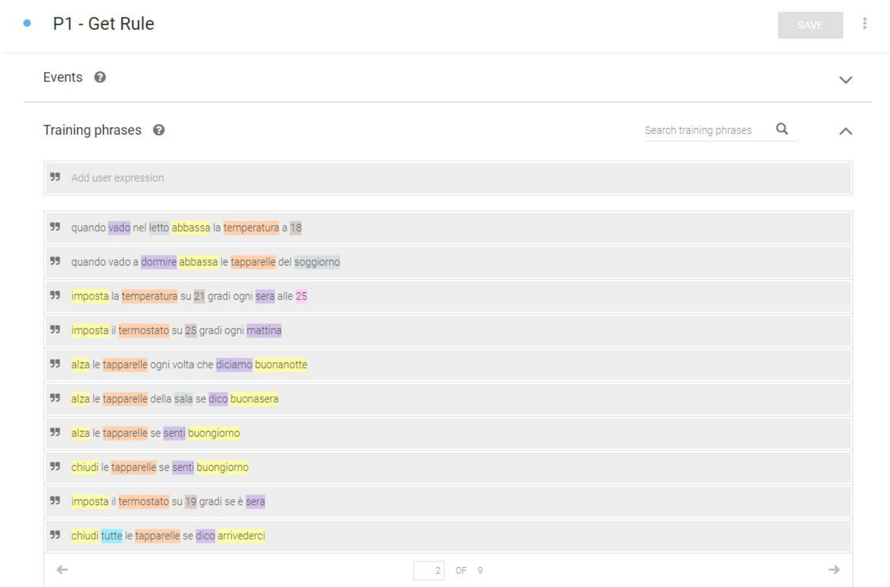


Figure 5.3: Training phrases of "P1 - Get Rule", the Italian phrases above are some of the utterances that a user should pronounce to create a rule.

- **Parameters**: describes the elements that are extracted from the user's sentence each time intent is found. The most used parameters are the entities, they correspond to structured data and are managed by the webhook; I analyzed entities in detail below in a dedicated section (Subsection 5.2.3). It is possible to define *"required"* parameters in order to force the user to insert those that are needed for the intent, if the agent doesn't found a required entity in the user expression, it explicitly has to request it. The Figure 5.4 shows all the parameters for the *"P1 - Get Rule"* intent, they correspond to all the entities needed for the rule creation.

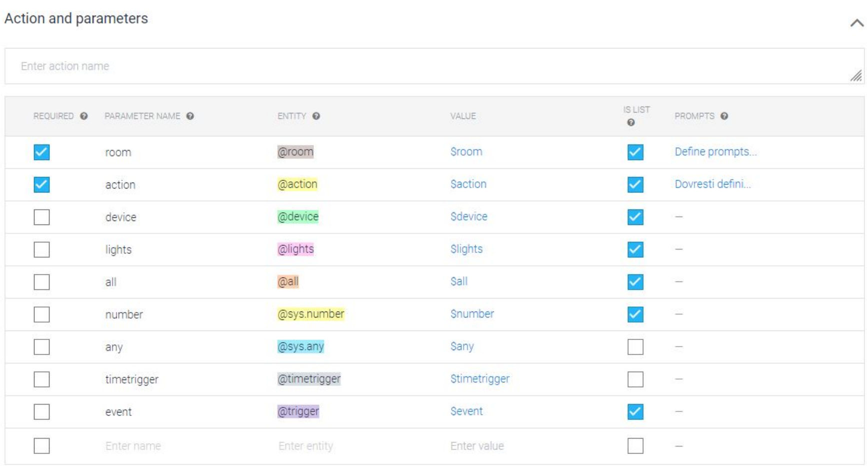


Figure 5.4: All the parameters of the rule creation for the Prototype 1, always referred to the intent *"P1 - Get Rule"*.

As also shown from the Figure 5.4, I chose as required entities the "action" (for prototype 1) and the "room" (for both prototypes). If the user does not define the action in a trigger-action rule, by pronouncing a rule with only the trigger, the action must be explicitly asked to continue the conversation. The same for the room, to be identified many IoT devices must have the indication of the room where are in.

- **Replies**: directly on the Dialogflow console GUI it is possible to define simple answers. These replies are pronounced by the agent in order to require further information from the end-users or interrupt the conversation. But since in the two prototypes the management of the intents is complex and requires queries to the database and particular algorithms, the management of the answers is postponed to the backend.

To activate the fulfillment for each Dialogflow intent is needed to set the webhook to enable from the Dialogflow console (Figure 5.5).

In this way the answer defined in the Dialoglfow console will not be pronounced by the agent but the system will always forward the request to the REST API of the NodeJS backend.

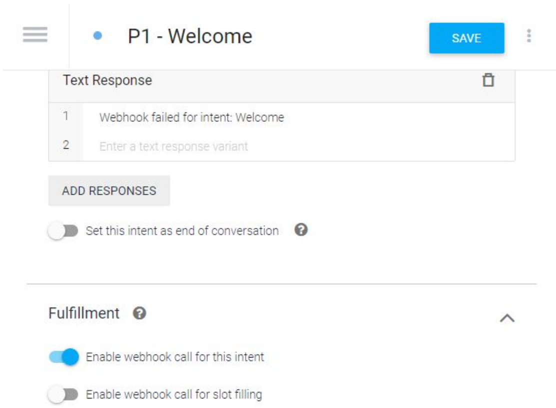


Figure 5.5: Enabling the webhook call for the intent "P1 - Welcome".

5.2.3 Entites

The above-mentioned parameters comprise a type called entity, the entities represent the main data extracted from the user's expression. Dialogflow provides system entities, otherwise, it is possible to create custom entities. The system entities that are used in the prototypes are *sys.any* and *sys.number*, the first allows the user to define a *"voice-command"* as a trigger to the action, the second one is used to capture the number related to the temperature. In our conversational agents, I dedicated particular focus on the entities of the *"P1 - Get Rule"* (prototype 1) and *"P2 - Get Event"* (prototype 2) intents. The agents should try to capture the entities that comprise which trigger, which device and which room make up the rule, and if they are not present, ask users for more details. In Figure 5.6, I tried to summarize all the entities used in Prototype 1, the same can be applied to Prototype 2, just by considering only the trigger section.

- **@trigger / @timetrigger**: specify the entities aimed at recognizing the trigger in the user expressions, the part of the rule that allows triggering the action.

- **@room**: each rule must have an associated room, the parameter is set as required. From the backend I have managed the explicit request to the user in case the @room entity is not present in the JSON file of the answer.

- **@device**: aimed at recognizing all smart devices of the house, except lights. During the development phase, I preferred to separate the devices from the lights because the latter could be of different types: bedside lights, ceiling lights, sink LEDs. The @device entity identifies when the expression contains

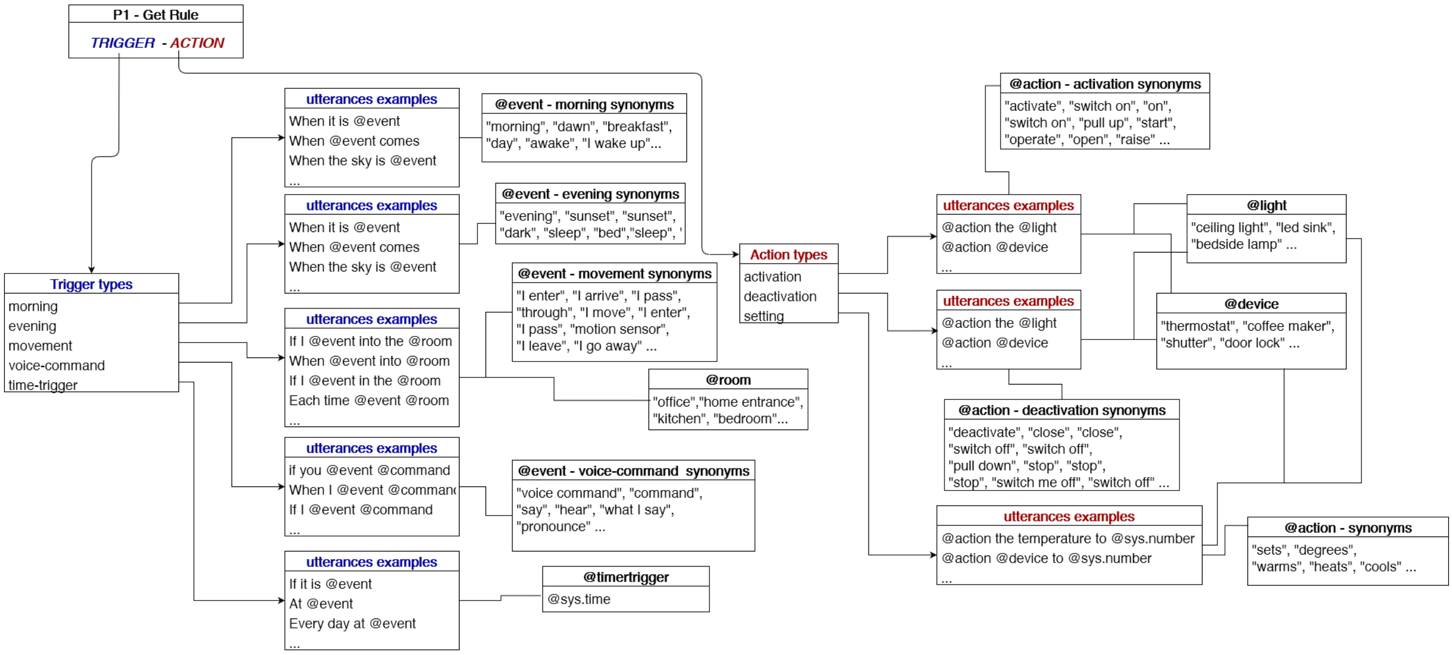


Figure 5.6: Entities management of the intent "P1 - Get Rule", the main operation of the full-vocal prototype (the second prototype has not the management of the action).

"shutters", "coffee machines", "thermostats" or "door locks", and all the possible synonyms in which the user can pronounce them.

- **@lights**: defines all types of lights that are present in the home scenario.
- **@action**: specifies all the possible actions that are connected to the smart devices defined in the home scenario. The actions are necessary for the completion of the rule. In fact, in Prototype 1, they are required parameters, so when they are not pronounced the backend will handle the request of the missing entity.

  In Prototype 2, the action is defined with the physical interaction of the user on the device. Therefore, it is not an entity used in the intents of the second prototype. The action management takes place through the interactions of the simulated devices, see the Section 5.5.

Some intents such as *"P1 - Lights disambiguations"* (from Prototype 1) are unnecessary for Prototype 2, because the manual action of the user on the desired device eliminates any kind of disambiguation, for example, if the user turns on the living room light, there is no reason to ask for further information.

The developed interfaces will call the webhook for most intents. The backend webhook is able to activate the function corresponding to the intent and manage the entities based on those received and those missing. Now, I analyze all the backend components of conversational agents.

5.3 Agents Backend

An alternative to using Dialogflow inline editor is a local machine, for having greater management and better control over the handling of intents. In order to do this, I enabled from the Dialogflow console the webhook function, so all intents, with the enabled function, can be managed through the webhook. The console needs the public URL of the webhook which through HTTPS Dialogflow will take care of sending the intents and receiving the generated replies. I used for deployment the serverless Google Cloud Functions for Firebase, while for the local deployment and the local testing I used a self-hosted and self-managed environment.

Firebase was used to set up project folders, for creating a Node.js environment, and for using the Firebase CLI to distribute the system at runtime. To initialize the project I followed the guide of Firebase [39], starting with login and authentication, I subsequently initialized all. After that, the skeleton project structure was created with the installation of all the *npm* dependencies. For both prototypes, I used the same Node.js environment and same backend webhook, in order to have one single ecosystem that could allow an agile production and evaluation.

5.3.1 Testing and Deployment

I used a local server such as express or the service provided by firebase. In order to allow fast iterations and testing with local hosting. However, Dialogflow needs a public and encrypted HTTPS server to access and make requests to the webhook, so I had to expose the webhook through a tunneling service. *Ngrok* was the service used, it allows to provide an HTTPS URL for the local webserver.

After initial local testing, I applied the distribution via Cloud Functions for Firebase. The Firebase deployment allows the creation of a public URL, no more local machine is needed, the entire Nodejs is public and ready to export all the requests and responses to the Cloud Functions generated URL.

5.3.2 Nodejs Webhook

I have created a webhook using JavaScript programming language also for the interaction with Dialogflow and for the possibility to use the client library of Actions on Google Node.js [40]. The library is installed via *npm*, the main package manager for Node.js.

The library allows to process user requests and send responses to Actions on Google for final user interaction. In short, the library represents the API for Dialogflow and Google on Actions services.

For the management of the service, I used an instance of a global object that allows managing the requests and responses, as already introduced, sent via HTTP requests in files in JSON format. The files contain both the archived data of the conversation and, those extracted from the intent.

Intent handler

Intent management takes place thanks to the *"Conversation"* object available for the aforementioned object instance, named *"app"*. The intent handling takes place thanks to functions called within managers of each intent.

Every time the webhook is called, an instance of the *"app"* object is generated, this is instance is responsible for the processing of the HTTP request, all the management occurs with a Javascript function like:

```
app.intent('intent name', conv =>
  //intent management
);
```

To send the response back to the user, the intent management needs to use the *ask()* function of the conversation object. So, by writing the responses in this function, the agent automatically receives back what to say. The library serializes the response, as in the request, in a JSON file as a payload of an HTTP request. Instead, to close the conversation, the *close()* function is used. Similar to *ask()*, it sends the response to the user but immediately afterward closes the application without listening the user answer.

5.4 Database

The database was needed to maintain the status of all the devices present in the simulated house, so the conversational agent can know the location, the device name set by the user, and its status. Since it is not the focus of the thesis, the management of adding devices by the user has not been covered. For this, I have preloaded the DB with a JSON file containing all the information about each devices such as the initial state, the id, the name, and the room in which are located. The Figure 5.7 shows a section of the stored information in the DB.

I selected the device's features from Google standards for Google Assistant applications [41]. All the device types allow applications to support a smart home environment. I choose to select some products from the Google lists, in particular, I

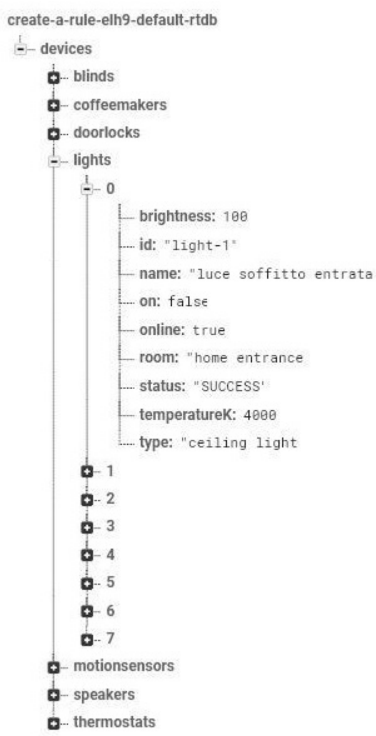


Figure 5.7: A representation of the DB preloaded on the Firebase Realtime Database.

selected the more recurring devices identified in interviews and in initial researches. The Figure 5.8 shows an example of a device used in the thesis agents, from the Google devices list. As also seen in the Figure 5.8, the attributes of each device can be recommended or required, I chose to provide users with only the required ones, so in the DB was inserted just those.

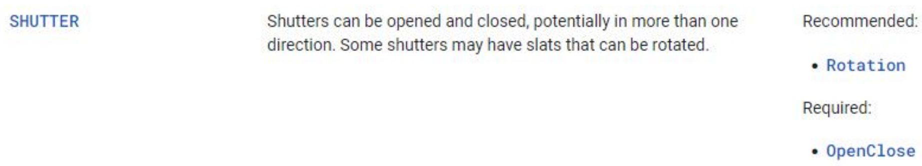


Figure 5.8: An element of Smart Home Device Types list in Google guide, the Shutters are one of the device type we choose to include in our simulated home scenario.

5.4.1 Query DB

The *iotAPI.js* project file (similar in both projects, many queries were the same) provides all the APIs that the backend uses to do dynamic discovery of the lists of a device type, such the functions *"getLights ()"* or *"getThermostats()"*. Those functions return the list of all devices in JSON format, through a *get* carried out on the DB.

To be managed and processed easily in JS, in the management of intents, the JSON files of the devices are used to build JS objects. The Figure 5.9 is an example of a Light object constructor.

```
class Light{
    constructor(id, on, brightness, temperatureK, room, name, type) {
      if(id) {
        this.id = id
      }
      this.on = on;
      this.brightness = brightness;
      this.temperatureK = temperatureK;
      this.room = room;
      this.name = name;
      this.type = type;
    }
}

export default Light;
```

Figure 5.9: Creation and initialization of a Light object

The DB is not used to save the rules; the projects focus on how to create these rules. For this reason, the loading of all the personalization rules is unnecessary.

The voice application of the Prototype 1 focuses the DB queries on the check if the user expression contains an existing device in the house. The system requires other queries in case the user doesn't specify a device in the room where the rule is composed. For instance, if a rule contains the switching on of a particular light, but the user forgets to specify it, the agent will query the DB to retrieve all the lights in the room where the user compose the rule.

The DB updates are limited to Prototype 2. If the user changes the state of a home device (e.g. by closing manually the shutters), the DB will be updated. Then, this changing of device state will be perceived by the Backend Node.js who send back a proper aswer. More details on communication with IoT devices are in Section 5.5. I stored information about user and conversation data in the userStorage field of the conversation object. The information contains the Action's memory associated with the user who uses the interface. If the assistant fails to match the identity of the user, the content is canceled. That was helpful to distinguish the users and their conversation, and also for identifying expert users or novel ones.

5.5 IoT Simulation

While the previous sections are similar for both prototypes, the second project requires special attention to the devices that need to be simulated. As already mentioned in the previous sections, the technology and architecture of a smart home are not relevant to the thesis, but it is necessary to mention how I chose to simulate the whole IoT ecosystem. Prototype 2 deals with simulating IoT controllers that allow the user to interact with simulated devices when this is expressly asked by the agent.

The simulated controllers are represented by switches, buttons, and selectors all displayed on a graphical interface of a website created for this purpose. In order to test Prototype 2, the device controllers are displayed on touch systems such as smartphones or tablets. The user interacting, for instance, with a light switch will trigger a change of attributes of the device field in the DB of the corresponding light. The change of state will be perceived by the NodeJS backend which will communicate to the conversational agent the device and the selected state.

The website was hosted on a public URL generated by Firebase, so a simple touch device with an internet connection could be used as the simulator of a switch in the test phase. The device controllers were displayed on different web pages, one for each device. At each launch of the website, each state of the IoT controlled was initialized by querying the DB.

Thanks to the graphic frameworks I have created a website with a navbar containing the links to the lists of all the devices with which the user can interact (I have not added the motion sensors, they are difficult to insert as an action of a personalization rule). The macro-categories of devices selected are: lights, shutters, door locks, thermostats and coffee machines. They are represented with icons with links to the lists of all the devices belonging to that category, for example, the Figure 5.10 shows the lists of all the lights in the simulated house. Furthermore, each element of the list is a further link to a dedicated page for each device, for example, I showed in Figure 5.11 the switch of a light. Before loading the switch page, a query is made to the DB to set the display on the last state of the device.

I generated the pages with React [42], a free JS library for building user interfaces components. The website created is a multi-page application that taking advantage of React, that combined with some toolkit as Boostrap and Material UI, provides multiple components that build a responsive and good-looking page design.

The agent will respond each time a user changes the state of a device in the proper turn of conversation. The backend section that manages the specific operation is present only for Prototype 2. The operations are managed by:

- **Intent *"P2 - Actions Listener"***: created in the Dialogflow console, it was used to capture the physical action of the user. The main operations, here

Figure 5.10: List of all the smart lights in the simulated house

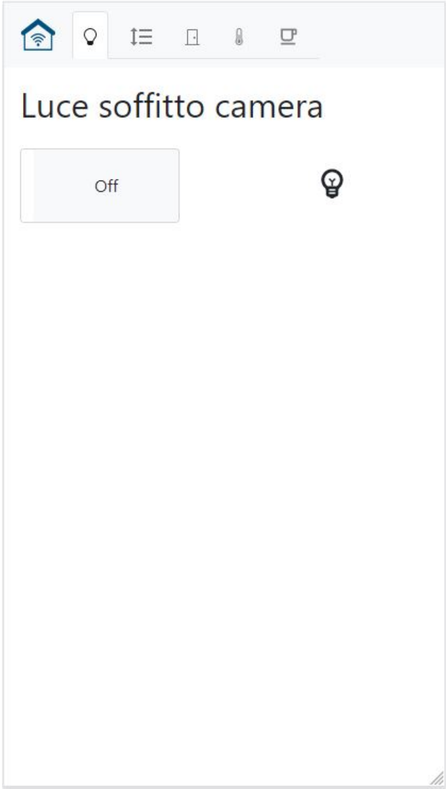


Figure 5.11: Simulation of light switch of central light in the bedroom

too, are postponed to the Node.js backend.

The intent is triggered every time the user confirms the event at the precedent request from the agent. So, the assistant will ask the user to act directly on the device which wants to add in the rule. In order to do that, the IPA requires a command to trigger the intent *"P2 - Actions Listener"*. I chose to make the user say "Modalità ascolto" ("Listen mode") to allow the agent to capture the user's action.

- **Intent management**: when the intent *"P2 - Actions Listener"* is matched, the IPA will switch to a waiting mode. This was the key of the operation, the assistant will wait until the user does the action required. By an *await* operation, the agent will wait for a *Promise* object, and the eventual completion of an asynchronous operation [43]. The total amount of waiting time possible is 9 seconds: the maximum time to wait for a response in the Google Assistant system. I have set 9 timeouts of 1 second that freeze the system, and after each second is checked if a state of a device is changed. If it happens, the system exit from a 9-second wait, and the rule can be completed. If the backend system not perceve the user action for 9 seconds, the agent will ask the user if wants to proceed with the “listen mode” or not.

- **Get the recent DB changes**: it is done in the *iotAPI.js* file, the async functions exploit the retrieving data of an asynchronous listener. Another important feature of Firebase Realtime Database is the retrieving of data by an asynchronous listener that is triggered any time the data changes [44]. This feature is well integrated with Node.js and I used it to wake up the agent from the 9-second “sleep” every time an attribute of a device changes in DB.

5.6 Action on Google

The Action represents the voice user interface of the agents, it is the one directly used by users to interact with the conversational interfaces. The main task of the Action on Google is the management and the sending of the input to the backend in order to elaborate the user expressions
Dialogflow allowed the direct integration with the Google assistant thanks to the built-in function. The section *"Integrations"* in the Dialogflow console, allowed enabling the Google Assistant integration, with the subsequent automatic construction of an Action. The operation was possible by choosing an *"Explicit Invocation"* which is the specification of the intent that is triggered when the user opens the Action, for example by saying *"Ok Google, talk to Create a Rule"*. Dialogflow, in this section, also allows integration with different textual interfaces such as Facebook manage Telegram or Slack, but these do not allow entirely voice communication and therefore were not used in this work. Taking advantage of Google Assistant allowed to translate the user’s audio into a string which was sent to the agent and subsequently to the backend. Likewise, it allowed the backend responses to be translated into audio via a text-to-speech system.

Google provides a simulator in the Action On Google console [45], the platform created by Google for developers to create Action on Google Assistant. The console allowed to simulate the interactions as if they were done with a real device with the Google assistant. I used the simulation to test both the conversational turns and any problems or bugs. The platform was also useful for choosing the name of the Action, and the type of assistant’s voice, it was also possible to manage the release phase to launch our application to Google Assistant users. For the application release, I completed the *"Deploy"* section by compiling the description, adding the image, and adding invocation phrases. These last were useful to define the phrases that the user can say to open the action, for example, in case *"Create a Rule"* as

the application name:
Ok Google, I want to create a rule.
Ok Google, can I create a rule?
Ok Google, let me create a rule.
I used different approaches in which the user could open the application, besides the classic *"Ok Google, talk to Create a Rule"*.

Voice

The interaction was conducted entirely through the conversations between the user and the IPA, so it was important to set a pleasant voice for the assistant and to manage the response sentences well. I used SSML (Speech Synthesis Markup Language) in the backend for the answers (generated by the above-mentioned *ask()* function). SSML helped to improve the text-to-Speech of the system, it allowed better customization in the audio responses. Details like the pauses were used to separate the sentences and give the right importance to the sentences spoken by the assistant. To set the break time of the assistant I used paused in speech synthesis, like: *<break time = "500ms" />*.

Alpha Release

I released the 2 prototypes in the Alpha channel and tested in the usability test described in the next Chapter 6. The Alpha Release allowed to distribute the Action to a small set of users without requiring a Google review.

Chapter 6

Evaluation

This chapter is dedicated to the evaluation of the prototypes in terms of usability with a particular consideration on the differences in ease of use and time efficiency between the 2 proposed prototypes.
The purpose of the test is to analyze whether the creation of trigger-action rules through conversation can make the personalization of smart homes easier and/or faster. The study will evaluate the 2 different approaches for the composition of rules: the completely vocal one and the other one based on physical action on home devices. Unlike graphical user interfaces tests, where the think-aloud methodology is often used, and people are asked to say what they are thinking as they complete tasks, this type of methodology cannot work with voice interfaces. In conversational agents test, think-aloud methods can have some undesired consequences, as an unwanted invocation of the application, or the introduction of wrong data inside the system. The solution is to ask some questions after each test task, in order to receive as soon as possible the real-time feedback of the participants.

6.1 Preparation and Setup

To perform the test, participants had to simply interact with the Google Actions developed on Google Assistant. For each participant, I have created a fake account that will serve to test the conversational prototypes and store the conversation history separately. The tests were carried out with one facilitator who will interact with the participant. The voice interfaces and the tests were created in Italian, the reason is the possibility of being tested on Italian soil and to facilitate the recruiting phase. More details on the localization of the prototypes have been specified in the Subsection 5.2.1.

At the beginning of the test, I provided users with the informed consent form and a pre-test questionnaire to learn more about the candidate's background. Furthermore, in the consent form, there was an explicit request on the recording agreement. The audio recording was required because taking many notes during the test could involve the participant less, the facilitator therefore always be available to the user who performs the test.

Participants have had to talk to the assistant in the most natural way possible, they were provided with a paper with the list of smart devices in the simulated house and a house plan with the indication of the rooms and the device's position. The tasks were set out one after the other by the facilitator. For each one, the notes were taken and the voice recorded (after the consent).

6.1.1 Smart Home Simulation

The evaluations of the prototypes started from a very specific home automation scenario, with smart devices defined and positioned in predetermined rooms. Specifically, the tests were performed in participant's houses, before each performance of tasks I configured the test room by positioning the smartphones necessary to simulate the IoT devices. The choice was made because a smart home imagined for testing is impractical, it would be difficult to get so many IoT devices such as smart bulbs, smart shutters, thermostats, smart locks, etc.
So, the evaluation was placed in a possible smart-home environment but simulated.

To accomplish this, as already explained in the Chapter 4, I have implemented a web application used to simulate the various devices, that allowed users to change their state.
Only a subset of operations have been selected and simulated to make the test more intuitive and simple.

6.2 Participants

As already happened in the semi-structured interviews, I recruited 10 users for the usability test. It is important, in order to have reliable results, not to recruit people too far from the aspects identified by the target user.
Using a Google Form questionnaire similar to the one already used, I identified the demographic characteristics of the target user (I selected the recruited participants with similar characteristics to the ones chosen for the interviews). The demographic characteristics are summarized below:

- **Age**: 16-50
- **Study background**: not technical experts or IT programmers.

- **Home automation interest**: moderate interest.
- **Familiarity with voice assistants**: I selected approximately half of the participants with familiarity and half without, in order to have a balance in the users' results.
- **Level of technological skills**: some awareness of technological devices.

The choice of the target user is focused on participants with a moderate interest in home automation, because the involvement of users during the test was essential. A user uninterested in home automation or, in any case, not accustomed to the use of technological devices, may not be sufficiently enticed in interacting with an intelligent speaker, and therefore in creating the rules. I chose the user without a high technical background because I wanted to adapt the prototypes to the different contexts of use, also to make the voice interface as accessible as possible.

6.3 Controlled Experiments

The tests undertaken want to evaluate the usability of the 2 prototypes.

- **Independent Variable (IV)** - prototype selection. The first prototype allows the rule composition in a single phrase, the second one separates the creation between trigger and action. The action composition requires direct user interaction with the chosen device.
- **Dependent Variable (DV)** - Errors Metrics, Time on Task, Subjective Measures, and System Usability Scale. I detailed the metrics measured below.

I performed the within-subject design solution, which allowed participants to test both prototypes.

Participants can perform better on the second prototype they try, they can benefit from the fact that they have already practiced on the first one. To avoid running into this problem and minimize the overall effect, I divided the participants into two groups, one group (GROUP A) started with Prototype 1 while the other group (GROUP B) with Prototype 2.
Using Conterbalacing:

GROUP A	GROUP B
GROUP B	GROUP A

Table 6.1: Participant groups division

The vocal prototypes developed have required an evaluation in the lab, in particular, Prototype 2 requires direct user interaction with the chosen device, which needs the physical presence of the participant. So, remote testing was not possible. The remote test would allow greater flexibility by not allowing the participant to move and go to the test site. But, Prototype 2 forced the candidate to be on the test site. An advantage of lab tests was to have a dedicated space to run user tests, especially important in voice application to have an isolated and quiet environment. I chose to ask the participants questions after each task, these are called Single Ease Question (SEQ) which must be short and not interrupt the running of the test. In order not to make users forget the perceptions of each task, it is necessary to ask them to express their impressions as soon as possible, before the end of the entire test.

Using a System Usability Scale (SUS) questionnaire, I developed a final set of questions to measure the perceived usability of the system, and some general impressions.

6.3.1 Evaluation Metrics

I wanted to have both feedback on prototypes and a tasks comparison between the two prototypes. In this section, I analyzed the metrics that I evaluated during the tests.

- **Task completion**: the completion of a task occurs after the user has started and completed the creation of the rule by correctly selecting the trigger and the action and involving the devices and the room that compose it. In both conversational agents, the operations were completed when the rule has been successfully set.
- **Correct Reject Error**: occured when a participant said something that was correctly not matched, because it was not an expected response for that state.
- **False Reject Error**: occured when a user said something that should be handled by the assistant, but it did not (this is often caused by the inaccurate training of the input phrases, the developer in most case do not consider some possible phrases)
- **Mismatch Error**: occurs when the IPA did not correctly recognize what the user is saying. This problem is mainly caused by the NLP unit, ambient noise, or bad spelling.
- **Time on Task**: the actual time that a participant took to complete the rule.
- **Subjective Measures**: after each task, the participant had to answer brief questions to measure the degree of perceived difficulty (through SEQ)

- **System Usability Scale**: after the completion of the tasks with one prototype, the participants had to compile the SUS (System Usability Scale) questionnaire to evaluate the usability, SUS is not an absolute measure but is generally used to measure the usability and the satisfaction perceived by users.

All the metrics listed above were taken during the test execution, both from notes and recordings. The two reject errors results in utterances that the assistant rejects can be both critical error and non-critical, it depends if the app crashes or the error management does not consider some possibilities.
I have chosen not to simply measure the duration because it is not always an index of the users' satisfaction rate. If the user feels in control and not confused or frustrated, it may not be a bad thing. But in most cases, when replies are unnecessary, too long, or are listing too many options, could make the conversation long and confuse many users. Furthermore, if the management of errors is found to be simple and smooth for the user, the errors can not ruin the user's perception. Structure subjective questions at the completion of each task based on the Likert scale can underline aspects as the difficulties of a particular task, even if the user completes the task correctly and without errors.
All the metrics can better identify the usability problems of the agents and allow have a way to compare the two interfaces.

6.4 Tasks

In order to test the main functionalities of the prototypes, I presented each participant with a finite number of tasks. During the presentation of the tasks, the facilitator should take care of how describing the tasks, in order to show to the user only the objective of the activity and not the task itself, he should only try to describe how to perform them. In our case, in the task's presentation, the words necessary for the creation of the rule should not be provided. The user should be guided only by the voice application.
Voice interfaces present many dialogue path possibilities, and each path could potentially correspond to one task that can be selected. It is necessary to focus on the task that really could be used most, select a task like: *"Make the lights in the living room turn on every time you enter the bathroom"*, it is a possible operation but it is not an activity that is probably used. During the interviews and the definition of the requirements, I analyzed which processes and devices are in the minds and needs of the participants, these were really helpful in tasks decision. Participants were reminded that they were here to help improve the system and were asked to say what they sincerely thought.

During the problems that arose in performing a task, the participants were not interrupted, only in the case in which they were frustrated by the situation and could not move forward. Not immediately answer questions like: *"What should I do now?"*, *"What should I say?"* encourage the user to try to find a solution on their own. If, on the other hand, the problem was frustrating or the app freezes, the intervention of the facilitator was necessary. For a direct comparison between the two interfaces, all tasks were repeated for both prototypes, also to acquire useful comparable metrics.

In order to identify the participant in a real situation to solve, I provided users with an appropriate context of a situation where they, probable owners of a new smart home, were to personalize the IoT ecosystem.

I show the list of the tasks (the translated version) that the facilitator has presented to participants during the tests:

- **Task 1**:

  > *You are in the kitchen and the light switch is in an awkward position. For this reason, you want the light to turn on every time someone enters the kitchen. Now try to tell the smart speaker to set this behavior.*

- **Task 2**:

  > *It's winter, and you decide that in order to save money on your energy bill and sleep better, you want to lower the room temperature when you go to bed every night. Try setting the automation by choosing a temperature below 22 degrees.*

- **Task 3**:

  > *The smart lock allows you to seal the entrance door every time you press the small display on it, but sometimes you forget to do it. Therefore, you would like to have the front door locked every time you enter the house.*

- **Task 4**:

  > *Today you woke up because of the shutters you forgot to close the night before, so you don't want to think about closing the blinds in the room every time you go to sleep. Try to set a rule to do this, you are free to choose the time you go to bed.*

- **Task 5**:

  > *You are in the kitchen, and you want the coffee machine ready for you when you wake up.*

- **Task 6**:

  > *When you go to bed at night and you are lying down, you always forget to turn off the lights. In order to avoid this, ask the smart speaker in the bedroom to turn off the lights in the room when you say a certain command of your choice.*

Each task has a quick description of the motivations why someone should create that rule. This creates more contextualization of the tasks and provides users with a real situation to solve. I did not choose a particular order in the presentation of the tasks in terms of activities difficulty. The participants have used both prototypes to complete the same tasks, first all six with one prototype and then with the other.

6.5 Equipment

I used the evaluation in-lab approach, which required special equipment:

- Audio recording
- Smartphone / smartspeakers running an Alpha version of Actions to test
- Internet connection
- Smartphones positioned in house spots to simulate the devices
- House plan of the simulated scenario as the one in the Figure 6.1 (and a quick device legend)

The usability test script and a copy of the document provided to the user with the SEQ questions and the house plan are attached to the appendix section of the thesis (Appendix B).

6.6 Results

After introducing the participants to the test they were going to take part in, I asked them (I was the facilitator for the entire 10 tests) for permission to record

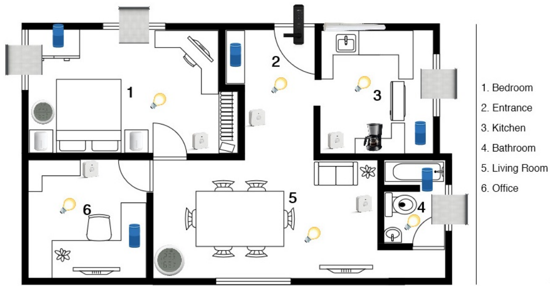


Figure 6.1: The house plan provided to users in the test

the audio session. The voice recordings were useful to accurately calculate the time on task and also to take note of the other evaluation metrics.

Before the actual start of the test, I provided the participants with a document containing the SEQs and the simulated house plan, as described in the Section 6.5. The tests carried out took longer than the estimated 15/20 minutes, they have lasted up to 30/45 minutes. Fortunately, this was not too frustrating for the participants, they stayed interested and involved in the tasks executions. Generally, they have participated with a moderate curiosity for the project, and at the end of the test, seemed involved and asked me multiple questions about the project's possible evolutions. As already happened in the user interview phase, I presented on the Table 6.2 the characteristics of the participants.

Participants and characteristics					
Participant	Age Range	Job sector	Familiarity with voice assistants	Home automation interest	Technological skills (avg)
P1	48-52	Health and Social care	✗	5	3.25
P2	53-57	Health and Social care	✓	4	4.0
P3	23-27	Health and Social care	✓	4	4.25
P4	23-27	Health and Social care	✓	4	4.5
P5	53-57	Housewife	✓	5	3.75
P6	53-57	Engineering and manufacturing	✓	5	4
P7	23-27	Health and Social care	✗	4	3.75
P8	23-27	Teacher training and education	✗	4	4.75
P9	23-27	Teacher training and education	✗	4	4.75
P10	23-27	Health and Social care	✗	4	4.5

Table 6.2: Demographic information of test participants

The participant's selection was made from users with a moderate interest in home automation. During the recruiting phase, those who did not show interest in this context were not selected, this choice was due to have good test involvement. The test was long and complex, a user not interested in the field would not have conducted the test with enthusiasm. The background questions were asked via a Google Form.

Among the 10 participants, none declared themselves an expert in computer programming, and school backgrounds and occupations were different. The heterogeneity of the participant was very useful because the application has been tested by user from different working sectors, from who work in construction to those in health care, from students to workers in the house.

In addition, some questions were asked regarding the use of voice assistants: if they thought they knew their main features and if they used them. The demographic survey also showed the self-evaluations of technological skills, as in user test (Chapter 3) the recruiting focused on users with a certain familiarity with technology. Thanks to the recordings and notes, I was able to measure the evaluations metrics described above (Subsection 6.3.1). These last sections describe the usability test results by comparing the metrics and calculate the usability score of the developed voice interfaces.

6.6.1 Quantitative Results

Execution Times

I chose to show first the results of the execution times of the tasks obtained, in Table 6.3 I represented all the tasks performed with both prototypes. For reasons of clarity, the analysis of time on tasks is separated from the others. Otherwise could be confusing to show all the evaluation metrics in a single graphical representation.

The first five participants represent the users belonging to the first group (who interacted with the full-vocal prototype first), while the last ones represent those who started the test with the second prototype.

Participant	$Task1$		$Task2$		$Task3$		$Task4$		$Task5$		$Task6$	
	Pr1	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*
P1	42s	74s	-	65s	60s	48s	-	45s	57s	62s	38s	64s
P2	41s	64s	-	53s	75s	50s	-	-	34s	56s	-	72s
P3	42s	73s	-	45s	25s	49s	-	64s	40s	53s	41s	45s
P4	32s	54s	-	60s	25s	37s	31s	54s	72s	55s	35s	45s
P5	58s	45s	-	54s	-	56s	-	61s	36s	51s	32s	60s
P6	35s	47s	31s	53s	-	47s	36s	58s	35s	42s	33s	110s
P7	46s	51s	58s	53s	25s	-	-	58s	35s	48s	35s	55s
P8	31s	47s	-	65s	26s	39s	-	72s	48s	47s	27s	54s
P9	29s	52s	36s	72s	38 s	47s	25s	72s	28s	78s	27s	43s
P10	41s	58s	30s	56s	25s	59s	29s	48s	29s	48s	31s	47s
Avg	**40s**	**56s**	**39s**	**58s**	**37s**	**48s**	**30s**	**71s**	**41s**	**54s**	**33s**	**59s**
Dev	22s	42s	25s	41s	24s	46s	25s	38s	27s	35s	25s	43s

Table 6.3: Execution times of the participants

Table 6.3 reports the execution times of the participants in the various tasks for each prototype developed. I also showed the times performed by the developer of the voice interfaces (Dev), allowing to show the *"optimal"* times on tasks that an expert user could tend to reach. Finally, I calculated the averages of the participants' activities (Avg).

The execution times have been calculated starting just after the IPA's welcome. The conversational agents developed present different introductions based on the number of times that a user opens the application, in any case, the welcome sentences are generated randomly. For this reason, and to have comparable data, it is more logical to start the time after the introduction.

Some tasks, such as Task 2 and Task 4, have been problematic for many users. They have been unable to complete successfully the task, these users continued until the deadline I set. I have chosen that after 120 seconds for Prototype 1 (Pr1) a user may feel too frustrating to continue and, in those cases, I have stopped the execution. For Prototype 2 (Pr2), I set a deadline of 180s. I estimated that the time is greater than the one before because the separation of the rule creation parts required more time. In fact, as expected, the times performed by the developer were lower for Pr1, and almost duplicated for Pr2. The data is due be to the fact that Pr2 requires creation of the rule in two phases, the first of the creation and the confirmation of the trigger and the second of the physical act on the device. These steps require greater execution times, in optimal conditions (without errors).

The same data was found with the participants, the average time of a single task completed by the participant was greater in Pr1 than Pr2. But the failure rate in the execution of some tasks, like in Task 2 and in Task 4, was very high. These tasks have a certain inherent difficulty, they require the creation of a rule with many details, in particular, the user should set the thermostat of a specific room to a certain degree and the rule should be activated each day at a certain time of day. The many parameters were confusing for most participants. They had to provide much information in a single sentence, and most of them had chosen words that had not been recognized from IPA. In other cases, they had used intricate phrases, difficult to understand from the IPA; the speaker did not could react by correcting users and leading them towards correction. I can say the same for Task4.

The Table 6.3 shows many problems with Pr1. Some users failed one and, sometimes, two or three tasks in single test session. However, this did not happen with Pr2, which was problematic only in 2 cases (P7 in Task 3 and P2 in Task 4).

Anyway, the results were satisfactory given that the participants did not follow a particular training on the trigger action rules; they did not know what grammar commonly used (if-then).

Among completed tasks, execution times with different participants were often variable for the same task. The reformulations and corrections of the sentences are the main reasons of this time disparity in the Table 6.3. Many users, guided by the application, had reformulated the rule. In some cases, IPAs did not understand the user's will, but thanks to the asking for rephrasing, participants had solved many problems in the rule creation.

I noted that the division of the participants in two halves (who start with Pr1 and who with Pr2) did not affect the execution times, and neither, as showed later, the usability evaluation.

On the other hand, a certain correlation can be noted between time on task and the self-evaluation of technological skills: users who were particularly expert in technology (<4.5 avg) got low times. They could complete the tasks with excellent times; the values are very close to those of the developer. This showed that if a user understands well how to create rules can quickly achieve excellent results. In the Table 6.4, I have shown the times obtained considering only the self-evaluated technology experts.

Participant	$Task1$		$Task2$		$Task3$		$Task4$		$Task5$		$Task6$	
	Pr1	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*
P4	32s	54s	-	60s	25s	37s	31s	54s	72s	55s	35s	45s
P8	31s	47s	-	65s	26s	39s	-	72s	48s	47s	27s	54s
P9	29s	52s	36s	72s	38 s	47s	25s	72s	28s	78s	27s	43s
P10	41s	58s	30s	56s	25s	59s	29s	48s	29s	48s	31s	47s
Avg	**33s**	**53s**	**33s**	**63s**	**28s**	**46s**	**28s**	**61s**	**44s**	**57s**	**30s**	**47s**
Dev	22s	42s	25s	41s	24s	46s	25s	38s	27s	35s	25s	43s

Table 6.4: Times on Tasks of tech-savvy participants

The Table 6.4 shows how the participants obtained average times shorter than the total average times. However, the overall improvement remains moderate if compared to the total average (about 10% less).

Furthermore, many execution times are very close to those obtained by the developer. In general, the technology experts completed more tasks than the other group, but also in this subgroup of participants, there were cases of users who failed to complete the rule, as shown by the "-". On the other hand, some self-evaluated non-tech-savvy didn't have problems with the same tasks (as P6 in Task 4). The overall conclusion is that the main difference found in tech-savvy is the shorter time they took to create the rules.

Task Completions and Error Metrics

In this section, I consider the remaining metrics measured in tests, I will analyze the errors, the task completion, and the SEQs.

In the Table 6.5 I have shown the percentages of task competition for all tasks in the 2 prototypes. I considered the completion when a participant completed a rule with all the parameters requested by tasks.

Prototype	*Task 1*	*Task 2*	*Task 3*	*Task 4*	*Task 5*	*Task 6*	***Total***
Pr1	100%	40%	60%	50%	100%	90%	**73.3%**
Pr2	100%	90%	90%	90%	100%	100%	**95%**

Table 6.5: Tasks completion percentage

In the previous analysis of the execution times, the particularly problematic tasks were Task 2, Task 3, and Task 4, only for Pr1. The amount of information to be pronounced during the rule creation confused many participants and some of them gave up, and others used words that the agent failed to understand. Even with the corrections and advice provided by the IPA, someone failed to adapt his expressions similar to the given examples.

The same problem was not encounter in Prototype 2, which allowed the task completion in practically all cases. The split between trigger creation and action (the physically act on the device) allowed a higher task completion rate. In only two cases, the participants could not complete the tasks even with attempts at sentence reformulation, they were stuck in trigger creation. They didn't try to rephrase, as IPA advised, and they were stuck in a loop. In one case, however, the completion was not correct because the participant mistakenly turned on the light when the task was to turn it off.

As introduced in Subsection 6.3.1, I have chosen to analyze three main errors. In this way, I obtained a complete vision of the causes of tasks failures in completions or the general reasons that delayed the completions.

Sometimes mismatch error occurred, this is mainly due to google's Natural Language Processing (NLU) [46]. Overall, the NLU worked very well. There were only a few mismatch errors: two with P1, one with P2, and two with P5. But in no cases, this type of error has corresponded to a non-completion of the task. Thanks to the error recovery system and to the user repetitions, these errors did not have an impact on the tasks completions, but only marginally on the time on task.

Also, I noticed a certain correlation between the age of the participants and this type of error. No participant with an age group under 30 years made mismatch errors. This indicates how a young user, even if self-declared to be unfamiliar with voice assistants, has had a certain familiarity and precision in speaking to smart speakers.

During the test, I considered the Correct Reject Error when a user tried to pronounce sentences that could not be interpretable by the IPA. In case of rule creation, a Correct Reject Error occurred when a rule formulation was very far from the examples proposed or, in any case, the sentence was not interpretable as personalization rule. Participants have resolved most of these errors with rephrasing required by the IPA. In case the IPA did not understand the user intent, it asked to repeat the sentence by providing examples. Some participants faced with Correct

Reject Error abandoned the task after 3 attempts. They preferred to not continue because of frustration, they said to found the task too complex.

False Reject Errors requires a more in-depth analysis. These errors affected only Pr1 and many rules formulated by participants during Task 2 and Task 4 should have been understood from IPA. However, it didn't happen. By refusing the creation, the IPA asked to repeat and reformulate the personalization rule, but in some cases, participants said sentences that were incorrectly considered as wrong. I summarized the main reasons for False Reject Errors:

- **Poor training of the agents**: during the training of intent, in development, it was not created an exhaustive list of users' expressions. The developer should create a lot of sample dialogs and insert them in the system, especially for the intent aimed at the rule composition is important a good training.

  An exhausting Early-Stage testing can solve the problem. An example of a this kind of test is the Wizard of Oz test, where a human is "behind the curtain" and acts as the IPA. This helps to give the illusion to participants that they are interacting with a "computer". The test can provide a lot of information about both intent training phrases and new possible conversational paths. In this thesis context, good Early-Stage testing could cover the greatest possible number of ways in which a rule could be composed.

- **Few synonyms of entities**: I encountered a problem with the synonyms of the @action entity (detail above in Subsection 5.2.3). Some users chose some words that I not considered in the synonyms filling in the Dialogflow Define Synonyms field. For example, I did not consider the verb *"take down"* referring to shutters action, but only *"pull-down"* or *"close"* or *"roll"*. Sometimes the IPA did not map the unknown value (*"take down"*) to the reference value (@action) and the rule can not be composed.

In the test, participants who did False Reject Errors (e.g. P1 in Task 4 or P5 in Task 2) failed to complete the task, neither with reformulations nor by choosing different terms they solved the problems. That result shows how participants who thought that their rule was correct did not try to adapt to IPA. However, many False Reject Errors can be solved by adding more training phrases.

A positive outcome of the test is the IoT devices detection: in Prototype 1 the action is composed of a device and a status, both of them must be pronounced correctly. In no case, IPA had problems in identifying those, even if the device name did not exactly correspond to the one used by the user. For example, P7 did not specify the device of the Task 6 rule, he said: *"turn on the light in the bedroom"*, when the bedroom has more light type (bedside table and ceiling light). The IPA by a request such as: *"which light do you want to turn on: room ceiling light or bedside lights or all lights?"*, can solve the disambiguation problem. Pr2 didn't have this kind of disambiguation problem because the physical act on IoT devices removes any doubt. If the user turns on the living room light, there is no doubt on the state (on) or the device (living room light)

Both prototypes achieved excellent results when participants should define a rule that contains the activation of a motion sensor as the trigger. They did not have any kind of errors. Users have said expressions like: *"if I enter the kitchen"*, *"if the motion sensor in the kitchen is activated"*, *"when someone passes through the kitchen door"*, or *"every time someone passes by the kitchen"*. In all these examples, which I reported the translation, were completed successfully.

Perceived Difficulty

After each task, I asked the participants a SEQ similar to this: *"overall, how difficult or easy was it to complete the task?"*. The SEQ rating scale has 7 points from *"Very Difficult"* to *"Very Easy"*. The Table 6.6 shows all the results obtained.

Participant	$Task1$		$Task2$		$Task3$		$Task4$		$Task5$		$Task6$	
	Pr1	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*	*Pr1*	*Pr2*
P1	6/7	7/7	2/7	6/7	4/7	6/7	2/7	6/7	6/7	6/7	6/7	6/7
P2	6/7	7/7	3/7	7/7	4/7	7/7	1/7	3/7	6/7	7/7	3/7	5/7
P3	7/7	7/7	2/7	7/7	7/7	7/7	2/7	4/7	7/7	7/7	7/7	5/7
P4	6/7	7/7	3/7	6/7	2/7	6/7	7/7	5/7	7/7	7/7	7/7	7/7
P5	4/7	6/7	2/7	6/7	2/7	6/7	1/7	5/7	4/7	7/7	5/7	7/7
P6	6/7	5/7	6/7	4/7	1/7	6/7	6/7	4/7	7/7	7/7	7/7	4/7
P7	5/7	5/7	5/7	5/7	1/7	2/7	3/7	6/7	7/7	7/7	7/7	6/7
P8	7/7	5/7	2/7	4/7	7/7	7/7	5/7	4/7	7/7	6/7	6/7	7/7
P9	7/7	6/7	6/7	5/7	7/7	6/7	5/7	5/7	7/7	5/7	6/7	6/7
P10	6/7	4/7	6/7	3/7	7/7	6/7	6/7	7/7	7/7	5/7	7/7	6/7

Table 6.6: Single Ease Question results

Overall, the tasks were considered easy (5 /7 and 6 / 7) or very easy (7/7) in the cases where users could complete the task successfully. Times on tasks did not affect the SEQ score, if the participants completed successfully the tasks they found them easy or very easy. P2, P3, P8, and P6 preferred to give a better score to tasks completed with a low time, but even in the case of several attempts, the score only dropped from 6/7 to 5/7. Instead, P6 chose a score of 4/7 for Task 6, which he completed, but with several difficulties. In fact, the participant took 110s to complete Task 6, much higher than the 43s of the developer.

The tasks considered more difficult (Task 2, Task 3, and Task 4), as expected, are those with lower scores, from 1/7 to 3/7. On average, Pr2 was found to be easier than Pr1, as also reflected by the task completion rate that I found strictly correlated with this difficulty score.

Instead, there has been no particular correlation between the SEQ score with demographic information.

6.6.2 Qualitative Results

System Usability Scale Results

At the end of the tasks with a specific prototype, the participants filled out a System Usability Scale (SUS) questionnaire, which was useful for measuring the usability of the voice interfaces. The Table 6.7 shows the SUS results obtained from the performed 10 usability tests. In terms of effectiveness, efficiency, and ease

Participant	*SUS score*	
	Pr1	*Pr2*
P1	72.5	75
P2	52.5	67.5
P3	65	77.5
P4	75	72.5
P5	52.5	70
P6	65	62.5
P7	75	62.5
P8	75	72.5
P9	90	82.5
P10	92.5	90
Avg	**71.5**	**73.3**

Table 6.7: SUS Scores

of use of the two voice interfaces, the obtained averages of SUS score determines a “good” usability.

As the table Table 6.7 shows, the usability scores of the two prototypes are quite similar, despite the strong differences in the task completions. This shows how the difficulties of some tasks did not affect the perceived usability, even if some tasks were particularly complex.

Furthermore, the self-evaluated tech-savvy participants preferred the usability of Prototype 1, they commented that the immediacy and speed in creating the rules were preferable. These participants found the Pr2 too intricate and complex for most people. They responded positively to the question 7 of the SUS, *"I would imagine that most people would learn to use this voice interface very quickly"*.

Paradoxically, this was disproved by users with low technological self-evaluation. They showed a strong preference for the usability of the Pr2.

6.6.3 Discussion

Many participants found both applications pleasant to use. They wished to have such a system for controlling smart devices in their home, even the participants who had more problems completing the tasks. Especially Prototype 1 has shown particular interest, participants thought that a similar voice application could encourage less technical people to program scenarios in their homes. The tests were useful to highlight the problems of the applications in terms of low flexibility in understanding the user. However, the problems and error encountered by users did not lose their enthusiasm. That reinforces the idea that the interface can be a convenient and immediate personalization system, by an appropriate refinement of the conversational agents. All participants said to find comfortable a conversational approach to the task they completed.

The integrations of different technologies like Dialogflow, NodeJs, Action on Google, and Firebase Realtime Database, necessary to build the agents, have led to minimum system latency. It didn't create particular frustration for the user, but it is an aspect to consider with more complex agents.

Although the problems can be a great frustration for users, good training of the agents can reduce most of the errors. The two rules creation approaches are complementary and they have not found preferences by the users on average. Not tech-savvy users have preferred Prototype 2, they commented that the creation was easier and more guided and that it generated fewer errors over the other prototype. Creating the rule in a single sentence made them nervous because they had to say too much information in one expression.

Prototype 1 was more attractive to tech-savvy, the speed at which they created the rules (even in less than 30s) was a great advantage over the "slowness" of Prototype 2. Overall, both prototypes were valid solutions and they guarantee the creation of rules in a comfortable and easy-to-use way for most users, I can be satisfied with the work done.

Chapter 7

Conclusions and Future Work

This last chapter summarizes the entire work, reflecting on the results obtained. At first, I summarise the difficulties encountered with the project, then I present the overall conclusion and the principal contributions. Finally, in order to improve the systems discussed, the chapter takes a look at the potential future works.

7.1 Results and Limitations

By focusing on people without a technical background, the development of voice interfaces has allowed a comfortable tool for the creation of personalization rules in a smart home scenario. This is a confirmation of how the conversational approach is a brilliant choice for IoT interaction. The study went through an initial literature review, continuing with the interviews, and the implementation, then finishing with usability tests. All these steps have been useful to identify that the end-users are well disposed towards the rule creation by conversing with smart speakers in an IoT ecosystem. An initial challenge was the choice of approaches to the rule creation by vocal. The interviews emphasized how the composition of rules was natural in trigger-action formalism. In the development phase, it was imagined a further solution that proposed a different approach: a rule creation divided into steps. This second approach, allowed to have a novel way to complete the rule, in particular, the action is created with the IPA asking the user a physical action on the home device, in order to show the IPA what to do.

One difficulty was the first idea of using physical smart devices. There are many IoT devices on the market, and each has different communication protocols. This can lead to great complexity in the management tools. So, I chose to simulate all the IoT devices that I used, in order to base the work on the focus of the thesis:

the approach to the creation of personalization rules.

A second problem was the asynchronous management of the input in Prototype 2, in particular, the control of the IPA activation as soon as a device was operated. It has been complex, but as highlighted by the usability test, it was quite functional.

Instead, the major problems found in the usability test were related to the difficulty of managing natural language. It was difficult to imagine how a complex command, as a personalization rule, could be said by people with different backgrounds and experiences.

Using different technologies, such as Dialogflow, NodeJs, Action on Google and Firebase Realtime Database has created minimal latency in the system, which has made the interaction less pleasant and fluid. A dedicated tool for smart speakers could provide the flexibility required in the rule creation process.

This thesis expands the works in the literature by focusing on the communication between the IPAs and the users for the personalization smart home environment. This study attempts to extend the current limited state of the smart speakers' capability, by creating new ways of interaction that could exploit better their potentiality. Nowadays the smart speakers are generally considered simple entertainment items. Novel approaches to creating even complex scenarios can have a certain impact on the home environment possibility of personalization.

Even if natural language is the biggest problem of the developed voice applications, the conversational approach has produced great feedback, but, however, it requires careful conversational agent training, by considering as many as possible of user expressions.

7.2 Future Work

Future work could expand the functionality of the developed voice applications, for example, by allowing users to configure even more complex rules. The addition of multiple actions, or the showing all the already set rules in a room, can be two futures implementations. If there are many rules, it is necessary to evaluate with which approach is better to show all the lists; the assistant listing all the rules one after another could become frustrating.

Furthermore, the usability test has given multiple examples and identify problems, in particular, in Prototype 1. Such information can be really helpful in refining the training of the agents, in order to consider as many queries as possible. This can be critical and improve the interaction experience of future users.

Subsequent work extends the evaluation by integrating real IoT devices into an existing scenario. This would generate more reliable usability tests and would allow a better evaluation of Prototype 2 in terms of device activation directly interacting with real IoT devices. However, this project idea requires a considerable development effort because considers real IoT devices of different brands and communication protocols, all integrated into a single system, which should be reactive and reliable.

Further reasoning on the study could take into account other novel approaches to the personalization of smart homes through conversation. An idea is the possibility of setting home scenarios through the previous configuration of all IoT devices in the desired states, and the possible activation of this scenario through a specific voice command. For example, the user could set the living room at a certain temperature, playing music at a predetermined volume, close the shutters and turning on some lights, and turning off others; then, the user could tell the IPA to set the scenario every time someone specifies a command like *"good evening"*. This approach to smart home personalization via the creation of complex scenarios is an alternative to the classic trigger-action formalism that requires a determined grammar and a structure for creating rules. A pre-configured home scenario by the user, and the simple activation can make this alternative a valid one, always taking advantage of the potential of the conversational approach.

There are many possible developments and the positive feedback of the work done is a promising indication of the future of the studies in the field.

Bibliography

[1] *Global smart speaker market 2021 forecast*. en-US. URL: https://www.canalys.com/newsroom/canalys-global-smart-speaker-market-2021-forecast?time=1616172721 (visited on 03/19/2021) (cit. on p. 1).

[2] Hsiao-Chi Ling, Hong-Ren Chen, Kevin K.W Ho, and Kuo-Lun Hsiao. «Exploring the factors affecting customers' intention to purchase a smart speaker». eng. In: *Journal of retailing and consumer services* 59 (2021). ISSN: 0969-6989 (cit. on p. 1).

[3] Pascal Kowalczuk. «Consumer acceptance of smart speakers: a mixed methods approach». eng. In: *Journal of research in interactive marketing* 12.4 (2018), pp. 418–431. ISSN: 2040-7122 (cit. on p. 1).

[4] Barbara Rita Barricelli, Elena Casiraghi, and Stefano Valtolina. «Virtual Assistants for End-User Development in the Internet of Things». eng. In: *End-User Development*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 209–216. ISBN: 3030247805 (cit. on p. 1).

[5] Roy Want, Bill Schilit, and Scott Jenson. «Enabling the Internet of Things». In: *Computer* 48 (Jan. 2015), pp. 28–35. DOI: 10.1109/MC.2015.12 (cit. on p. 2).

[6] Andrea Newman. 1976. URL: https://developer.amazon.com/alexa (cit. on p. 2).

[7] URL: https://assistant.google.com/ (cit. on p. 2).

[8] URL: https://www.apple.com/siri/ (cit. on p. 2).

[9] John Karat. *End User Development*. eng. Springer Netherlands, 2006. ISBN: 1-4020-4220-5 (cit. on p. 2).

[10] Barbara Rita Barricelli and Stefano Valtolina. «Designing for End-User Development in the Internet of Things». eng. In: *End-User Development*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 9–24. ISBN: 3319184245 (cit. on p. 2).

[11] Daniela Fogli, Rosa Lanzilotti, and Antonio Piccinno. «End-User Development Tools for the Smart Home: A Systematic Literature Review». eng. In: *Distributed, Ambient and Pervasive Interactions*. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 69–79. ISBN: 331939861X (cit. on p. 2).

[12] Yinbo Yu and Jiajia Liu. «TAPInspector: Safety and Liveness Verification of Concurrent Trigger-Action IoT Systems». eng. In: (2021) (cit. on p. 2).

[13] IFTTT. *IFTTT*. URL: `https://ifttt.com` (visited on 03/19/2021) (cit. on p. 2).

[14] *Zapier | The easiest way to automate your work*. URL: `https://zapier.com/` (visited on 03/19/2021) (cit. on pp. 2, 5).

[15] *Power Automate | Microsoft Power Platform*. URL: `https://flow.microsoft.com/en-us/` (visited on 03/19/2021) (cit. on p. 2).

[16] *Create scenes and automations with the Home app*. en. URL: `https://support.apple.com/en-us/HT208940` (visited on 03/19/2021) (cit. on p. 2).

[17] *Node-RED*. URL: `https://nodered.org/` (visited on 03/19/2021) (cit. on pp. 2, 5).

[18] E.W Dijkstra, F.L Bauer, and M Broy. «On the foolishness of "natural language programming"». eng. In: *Lecture notes in computer science*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1979, pp. 51–53. ISBN: 9783540092513 (cit. on p. 4).

[19] Michal Gordon and David Harel. «Steps towards Scenario-Based Programming with a Natural Language Interface». eng. In: *From Programs to Systems. The Systems perspective in Computing*. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 129–144. ISBN: 3642548474 (cit. on p. 4).

[20] Jessica Van Brummelen, Kevin Weng, Phoebe Lin, and Catherine Yeo. «Convo: What does conversational programming need? An exploration of machine learning interface design». eng. In: (2020) (cit. on p. 4).

[21] URL: `https://ifttt.com/` (cit. on p. 5).

[22] Amir Rahmati, Earlence Fernandes, Jaeyeon Jung, and Atul Prakash. «IFTTT vs. Zapier: A Comparative Study of Trigger-Action Programming Frameworks». eng. In: (2017) (cit. on p. 5).

[23] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. «A High-Level Semantic Approach to End-User Development in the Internet of Things». eng. In: (2019) (cit. on p. 5).

[24] Marco Manca, Parvaneh Parvin, Fabio Paternò, and Carmen Santoro. «Integrating Alexa in a Rule-based Personalization Platform». In: Sept. 2020, pp. 108–113. DOI: 10.1145/3411170.3411228 (cit. on p. 5).

[25] Anoja Rajalakshmi and Hamid Shahnasser. «Internet of Things using Node-Red and alexa». eng. In: *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*. IEEE, 2017, pp. 1–4. ISBN: 1509065148 (cit. on p. 5).

[26] Ravi Kishore Kodali, Sasweth C Rajanarayanan, Lakshmi Boppana, Samradh Sharma, and Ankit Kumar. «Low Cost Smart Home Automation System using Smart Phone». eng. In: *2019 IEEE R10 Humanitarian Technology Conference (R10-HTC)(47129)*. IEEE, 2019, pp. 120–125. ISBN: 9781728108346 (cit. on p. 5).

[27] Luigi De Russis and Alberto Monge Roffarello. «Personalizing IoT Ecosystems via Voice». eng. In: CEUR-WS, 2020 (cit. on p. 6).

[28] Ting-Hao Huang, Amos Azaria, Oscar Romero, and Jeffrey Bigham. *InstructableCrowd: Creating IF-THEN Rules for Smartphones via Conversations with the Crowd.* Sept. 2019 (cit. on p. 6).

[29] Fulvio Corno, Luigi De Russis, and Alberto Monge Roffarello. «HeyTAP: Bridging the Gaps Between Users' Needs and Technology in IF-THEN Rules via Conversation». eng. In: ACM, 2020 (cit. on p. 6).

[30] Evropi Stefanidi, Maria Korozi, Asterios Leonidis, and Margherita Antona. «Programming Intelligent Environments in Natural Language: An Extensible Interactive Approach». eng. In: *Proceedings of the 11th PErvasive Technologies Related to Assistive Environments Conference.* PETRA '18. ACM, 2018, pp. 50–57. ISBN: 9781450363907 (cit. on p. 6).

[31] André Sousa Lago, João Pedro Dias, and Hugo Sereno Ferreira. «Managing non-trivial internet-of-things systems with conversational assistants: A prototype and a feasibility experiment». eng. In: *Journal of computational science* 51 (2021), p. 101324. ISSN: 1877-7503 (cit. on p. 6).

[32] Julia Brich, Marcel Walch, Michael Rietzler, Michael Weber, and Florian Schaub. «Exploring End User Programming Needs in Home Automation». eng. In: *ACM transactions on computer-human interaction* 24.2 (2017), pp. 1–35. ISSN: 1073-0516 (cit. on p. 7).

[33] Allen St. John. *speakers.* URL: https://www.consumerreports.org/smart-speakers/smart-speakers-are-your-gateway-to-the-smart-home-a9720483807/ (cit. on p. 7).

[34] *Video conferencing, Cloud Phone, WEBINARS, Chat, Virtual EVENTS: ZOOM.* URL: https://zoom.us/ (cit. on p. 8).

[35] *Google forms - create and analyze surveys, for free.* URL: https://www.google.com/forms (cit. on p. 9).

[36] *Conversation design | google developers.* URL: https://developers.google.com/assistant/conversation-design/ (cit. on p. 20).

[37] Wilhelm Schneider-Clauss and Heribert A. Hilgers. *Alaaf Kölle: En Schelderei Us grosser Zick*. 1980. URL: https://developer.amazon.com/en-US/docs/alexa/custom-skills/voice-design-best-practices-legacy.html (cit. on p. 20).

[38] Cathy Pearl. *Designing voice user interfaces: Principles of conversational experiences*. O'Reilly, 2017 (cit. on pp. 20, 26).

[39] *Get started: Write, test, and deploy your First Functions | Firebase.* URL: https://firebase.google.com/docs/functions/get-started#set-up-nodejs-and-the-firebase-cli (cit. on p. 40).

[40] *Build fulfillment with the actions on google node.js client library (Dialogflow).* URL: https://developers.google.com/assistant/df-asdk/reference/nodejsv2/overview (cit. on p. 40).

[41] *Smart home device types | actions on google smart home.* URL: https://developers.google.com/assistant/smarthome/guides (cit. on p. 41).

[42] *React – a JavaScript library for building user interfaces.* URL: https://reactjs.org/ (cit. on p. 44).

[43] *Promise - javascript: Mdn.* URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise (cit. on p. 45).

[44] *Retrieving data | firebase realtime database.* URL: https://firebase.google.com/docs/database/admin/retrieve-data?hl=en (cit. on p. 46).

[45] URL: https://console.actions.google.com/ (cit. on p. 46).

[46] *Natural language processing.* URL: https://research.google/research-areas/natural-language-processing/ (cit. on p. 59).

Appendix A

User Study Interview

The appendix presents the Italian version of the user study in form of an interview. These questions allowed us to lead the semi-structured interviews.

A.1 Quick introduction

Faccio parte di uno studio di ricerca volto ad analizzare le possibilità di personalizzazione di un ecosistema IoT attraverso la conversazione. Ti farò alcune domande sulla automazione domestica, successivamente ti chiederò di reagire onestamente ad una situazione ipotetica. Volevo rassicurarti sulle domande, nessuna risposta può essere sbagliata, non preoccuparti!

A.2 Background Questions

1. Puoi descrivere la tua esperienza con gli smartspeakers? Se non hai mai direttamente interagito con questi, quali sono le principali ragioni che non te ne fanno possedere uno?
2. (Smart speakers users)
    - Mi descriveresti gli utilizzi che fai del/degli smartspeaker che possiedi?
        - Quale funzionalità del prodotto utilizzi di più? Perchè?
    - Quali sono le principali difficoltà che hai riscontrato nell interazione con gli smart speakers? Hai mai avuto difficoltà a fare capire le tue intenzioni all' assistente, in che contesti?
    - Raccontami la tua esperienza con i dispositivi IoT come lampadine illuminazione smart, termostato intelligente, tecnologia per la casa pulita,

interruttore smart etc..? Ne possiedi qualcuno? Quali utilizzi in connessione con l'assistente vocale?

  – Come pensi che amplierai il tuo ecosistema IoT?

3. (Not Smart speakers users)

   - Pensi di conoscere le potenzialità degli smartspeakers? Potresti raccontarmi cosa conosci?
   - Cosa ne pensi dei disposivi IoT come lampadine illuminazione smart, termostato intelligente, robot per la casa pulita, interruttori smart etc..? possiedi/hai intenzione di possederne qualcuno?

4. Dovessi immaginare una casa intelligente, con molti dispositivi smart e smartspeakers, che processi automatizzeresti e quali no?

5. Sei consapevole che gli SmartSpeakers stessi offrono la possibilità di configurazione delle regole ad es.
   1) Quando l'allarme si spegne, accendi la luce e riproduci le informazioni meteorologiche.
   2) Tutti i giorni alle 24 spegni tutte le luci e chiudi tutte le persiane e la porta.

   - (Se si) In che contesto ne hai sentito parlare?
     – Hai mai provato a crearne qualcuna?
     – Che piattaforma hai utilizzato?
     – Quali problemi hai riscontrato durante la loro creazione?

6. Che idea hai di questi processi automatizzati direttamente configurabili tramite l'autoparlante intelligente?

A.3 The Imagination Exercise

Ora, voglio chiederti come reagiresti se interagissi con un assistente conversazionale - in particolare, ti chiederò di eseguire alcune operazioni, che ora ti elencherò. Premetto che non puoi fare o dire nulla di sbagliato, vogliamo solo che tu reagisca con il dispositivo nel modo più naturale e onesto possibile.

Supponiamo che tu vivessi in una casa automatizzata con sensori di passaggio all'ingresso delle stanze e dispositivi intelligenti come lampadine o elettrodomestici intelligenti.

Il tuo desiderio è quello di accendere la luce centrale della cucina ogni volta che entri in quella stanza

- Puoi mostrarmi cosa diresti al tuo altoparlante intelligente per impostare questa regola se sei all'interno della cucina e parli con l'assistente nella stanza?
- E nel caso tu fossi in camera da letto cosa pensi sia naturale dire per impostare la regola?
- Cosa ti aspetti che succeda dopo aver detto la regole?

Ti propongo un altro scenario possibile.
Vuoi impostare il fatto che quando vai a letto si chiudano le tapparelle di casa e si spengano tutte le luci

- Cosa diresti se tu fossi nella camera da letto per impostare la regola?
- Cosa invece diresti fossi in cucina?
- Cosa ti aspetti che succeda dopo aver detto la regole?

Riprendendo la situazione della prima regola, ti propongo una condizione in cui lo smart speaker ha diversi livelli di automazione e consapevolezza del contesto.
(I vari scenari saranno poi adattati ragionevolmente in base alle risposte precedenti del utente)
Ti presento quindi 4 scenari possibili, rispondi quale pensi che sia più naturale e cosa ti aspetti che risponda lo smart speakers:

1. Il dispositivo automaticamente setta la regola senza chiederti conferma, capisce di che stanza stai parlando e che dispositivi hai indicato e rispondendo ad esempio *"ok, regola impostata"*
2. L'assistente riformula la regola secondo la sua grammatica e te la ripete in modo che tu capisca se il dispositivo ha compreso tutto il necessario per applicarla ad esempio *"La regola appena impostata è riferita alla stanza "CUCINA", si accenderà la "LUCE CUCINA PRINCIPALE", se il "SENSORE DI MOVIMENTO CUCINA" viene attivato, corretto?"* [SI/NO].
3. Il dispositivo scompone il messaggio in blocchi singoli consentendo all' utente di disapprovarli ciascuno. Ad esempio *"La regola appena impostata è riferita alla stanza "CUCINA"?"*[SI/NO]; *"La luce che vuoi si accenda è "LUCE CUCINA PRINCIPALE"?"* [SI/NO] *"Il sensore di passaggio è "SENSORE DI MOVIMENTO CUCINA"?"* [SI/NO] e come nel caso 2 ti chieda una conferma ripetendo tutta la regola.
4. Il dispositivo capisce che quella stai dicendo è una regola e capisce il tipo di dispositivi vuoi interagiscano. Esempio: *"Una regola è composta da un evento e da un azione, che "SENSORE" vuoi funga da evento?" "Quale "LUCE"*

desideri che si accenda?" si tornerebbe poi a una situazione come la 2 o 3 per una conferma.

- Cosa ne pensi degli scenari sopra? Ne trovi alcuni confusionari, pensi sia difficile da sopportare una conversazione più lunga e complessa come negli ultimi casi?

- Nel caso in cui lo speaker non capisca automaticamente dal contesto il tipo di dispositivo come ad es *"Il dispositivo ha riconosciuto automaticamente "LUCE CUCINA PRINCIPALE" ma la tua intenzione era "LUCE CUCINA LAVELLO""* quale scenario ti aspetteresti:

  1. L'assistente ti elenca tutte le LUCI della casa connesse con l'assistente.
  2. L'assistente ti chieda prima in che stanza si trovi la luce indicata e successivamente elenchi solamente le luci della stanza indicata.
  3. L'assistente capisce che sei nella cucina e ti elenchi solamente le luci della stanza dove hai parlato, questo richiederebbe che tu fossi in quella stanza.
  4. Che tu per indicargli correttamente la luce la accenda manualmente per fargli capire cosa stai indicando.

- Cosa ne pensi di uno scenario del genere in cui non avviene nessun comportamento automatico? Pensi possa essere frustrante intraprendere una conversazione del genere?

Esempio di una possibile conversazione
- *Utente*: voglio impostare una regola
- *Assistente*: ok, quale evento vuoi accada?
- *Utente*: quando vado a letto
- *Assistente*: a che ora vai a letto?
- *Utente*: ore 22:00
- *Assistente*: L'evento sarà verificato all'attivazione del "ORE 20:00"? [si/no]
- *Utente*: si
- *Assistente*: quale evento vuoi che si verifichi?
- *Utente*: si spengano tutte le luci e si chiudano tutte le tapparelle.
- *Assistente*: L'azione è "Spegnimento di TUTTI i dispositivi di tipo "LUCE" e chiusura di TUTTI i dispositivi di tipo "TAPPARELLA"?"
- *Utente*: si
- *Assistente*: La regola appena impostata spegnerà tutti dispositivi di tipo "LUCE" e chiuderà tutti i dispositivi di tipo "TAPPARELLA", SE sono le ore 22:00, corretto? [si/no])

Appendix B

Usability Test

The appendix presents the Italian version of the usability test script, below a copy of the document provided to user at the test session

B.1 Instructions

Ciao, apprezzo molto che tu abbia dedicato del tempo della tua giornata per partecipare a questo test. Sono Carlo e sono uno studente del Politecnico di Torino.

Ora, lasciami dire come andrà. Vorrei iniziare ponendoti alcune domande su chi sei, sul tuo background e sulla tua esperienza lavorativa. Ti chiederò quindi di eseguire alcune attività utilizzando due applicazioni vocali. Una volta che queste attività saranno completati, ti farò qualche domanda più generale su quello che hai appena provato.

La sessione dovrebbe durare circa 15 minuti. La prima cosa che voglio chiarire subito è che stiamo testando l'applicazione, non stiamo testando te. Non puoi dire o fare nulla di sbagliato. Quindi non preoccuparti: non puoi fare nessun errore!

Ti chiederò di parlare e interagire nel modo più naturale possibile. Inoltre, non preoccuparti di ferire i nostri sentimenti. Lo stiamo facendo per migliorare le applicazioni, quindi abbiamo bisogno di ascoltare le tue reazioni oneste.

Alla fine di ogni attività dovrai rispondere a 2 semplici domande a risposta chiusa, che ti consegnerò a breve, inoltre se avrai domande sarà quello il momento di porgerle: tra un attività e l'altro.

Ora ti chiederò il permesso di registrare la sessione tramite un registratore audio: puoi farlo compilando questo modulo Google. Nello stesso modulo, troverai anche quelle domande su di te, di cui ti accennavo poco fa.

Hai domande ?

Questionario pre-test (link al modulo google del documento di consenso informato e al questionario, da fornire al partecipante).

Ora ti spiego brevemente la struttura e lo scopo delle applicazioni vocali che andrai a testare:

> *"Le soluzioni proposte da questa tesi analizzano le possibilità di creare regole, comportamenti di personalizzazione tra dispositivi smart in casa, tramite la conversazione. Sono stati sviluppate 2 applicazioni vocali che permettono di creare regole utilizzando 2 metodi differenti per la loro creazione. Queste regole, dette regole evento-azione, permettono il verificarsi di un'azione, come l'accensione di una luce, all'attivazione di un evento, come lo scoccare di una certa ora.*
>
> *Interagirai quindi con prototipi di un'applicazione in esecuzione su un dispositivo dotato di Google Assistant, in questo caso uno smartphone, data la mancanza di uno smart speaker google interagirai con uno smartphone utilizzandolo come un autoparlante, quindi ti chiediamo di non leggere o scrivere ma solo interagire vocalmente. Inoltre, quando aprirai l'applicazione, affronterai una prima introduzione che ne spiegherà la funzionalità, e poi risponderai e interagirai a voce con l'assistente. Le aperture successive dell'applicazione, per proseguire con alte attività, non avranno le istruzioni come al primo avvio"*

B.2 Test

Voglio ricordarti, che se qualcosa non funziona, sicuramente non è un tuo problema ma delle applicazioni. Non ci sono risposte sbagliate o niente che tu possa dire qui è errato. Ti chiedo solo di agire nel modo più naturale possibile, cercando di comportarti come se stessi provando l'applicazione in solitaria.

B.2.1 Context Presentation

Introduco il contesto dove dovrai immaginare di eseguire il test:

Sei madre (/padre) di una famiglia di 4 componenti: 2 figli maschi e il tuo compagno/a. Vi siete appena trasferiti in una casa con diversi dispositivi domotici:

tapparelle, luci, lampadine, serratura d'ingresso, termostati e una macchina da caffè, tutti dispositivi connessi e intelligenti. Ci sono anche altoparlanti intelligenti come un Google Home o un Amazon Echo in ogni stanza. Vorresti personalizzare il comportamento di alcuni dispositivi in casa, parlando con gli smart speaker, qui sono rappresentati dallo smartphone con Google Assistant.

B.2.2 Tasks

(Introduzione che recito all'inizio del solo prototipo che richiede l'azione fisica dell'utente)
L'assistente ti chiederà di interagire con i dispositivi della casa, che in questo caso verranno simulati tramite smartphone posizionati al posto degli interruttori dispositivi smart. Ad es l'interruttore della luce, il pulsante per alzare le persiane o il per chiudere la serratura di casa saranno sul display dello smartphone. Prima di iniziare ogni task posizionerò il telefono in corrispondenza della posizione del dispositivo richiesto dell'attività in modo che tu possa interagire con esso.

Ora ti porgo una scheda, sul retro è presente la piantina della casa con cui andrai ad interagire, ti sarà utile più che altro per ricordarti quali sono i dispositivi intelligenti,e la loro posizione nelle stanze, anche se questi verranno simulate. Mentre quelle che vedi in alto sono semplici domande che dovrai rispondere ogni volta che finirai un'attività. Ti chiederò di provare a svolgere alcune attività specifiche che leggerò ad alta voce, uno per volta.

TASK 1

Per avviare ogni conversazione e aprire l'applicazione vocale dovrai dire:
"Ok Google, Parla con Crea una Regola" (/ "Ok Google, Parla con Crea una Regola 2") (Nella versione alpha dei prototipi le personalizzazioni sulle frasi di invocazione sono limitate!)
Il primo compito:

> *Ti trovi in cucina e l'interruttore della luce è in una posizione scomoda. Per questo motivo desideri che la luce si accenda ogni volta che qualcuno entra in cucina. Ora prova a dire allo smart speakers (smartphone) qui di impostare questo comportamento.*

(se interazione prototipo 2, il facilitator indica al participant la posizione dello smartphone che rappresenta l'interruttore)

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?

TASK 2

Ora, stai interagendo con lo speakers in salotto:

(P2, il facilitator indica al participant la posizione dello smartphone che rappresenta il termostato)

> *È inverno e per risparmiare sulla bolletta e per dormire meglio vuoi abbassare la temperatura di del camera quando vai a letto ogni notte. Prova a impostare l'automazione scegliendo una temperatura a tua scelta sotto i 22 gradi.*

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?

TASK 3

Ora immagina di svolgere questa attività sempre parlando con lo smart speaker del salotto: (P2, il facilitator indica al participant la porta con appoggiato lo smartphone)

> *La serratura intelligente ti consente di sigillare la porta d'ingresso ogni volta che premi il piccolo display su di essa, ma a volte ti dimentichi di farlo. Quindi, vorresti avere la porta d'ingresso chiusa a chiave ogni volta che entri in casa.*

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?

TASK 4

Ora il quarto attività:

> *Parlando con lo speaker in camera da letto, vuoi non doverti preoccupare di chiudere le persiane della stanza quando vai a dormire. Imposta una regola per farlo, sei libero di scegliere l'orario in cui vai a letto.*

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?

TASK 5

La penultima attivtà che dovresti eseguire:

Ti trovi di nuovo in cucina, vuoi far sì che la macchina del caffè sia accesa per te quando ti svegli.

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?

TASK 6

Ora siamo alla fine.
L'ultimo attivtà consiste in:

Quando ti corichi nel letto, la sera, ti dimentichi di spegnere le luci,per evitare ciò, chiedi allo smart speaker in camera da letto di spegnere le luci nella stanza quando pronunci un determinato comando (a tua scelta)

(invita il partecipante a completare la SEQ)
Hai riscontrato problemi?
(Ripeti tutte le attività per l'altro prototipo)

Congratulazioni, hai completato tutti le attività!!!

Ti chiediamo di compilare un Questionario Post-Test che ci permetta di misurare l'usabilità percepita.

Ora che abbiamo finito, hai qualche domanda?

B.3 User Document

TASK 1 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 1 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 2 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 2 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 3 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 3 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 4 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 4 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 5 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 5 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

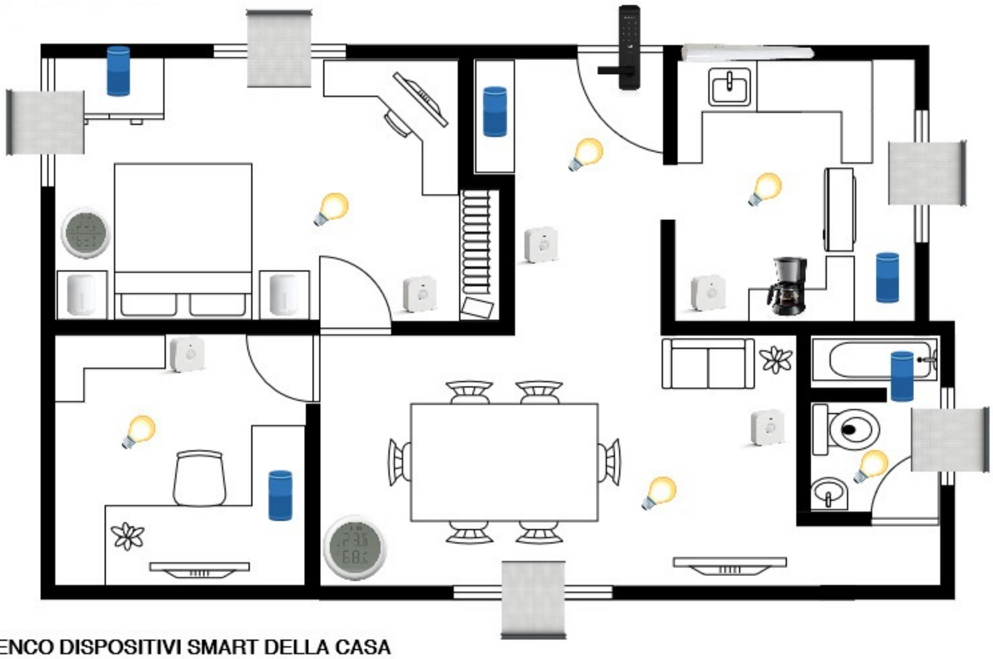
TASK 6 - P1

Nel complesso quanto hai trovato difficile o facile questo compito?

TASK 6 - P2

Nel complesso quanto hai trovato difficile o facile questo compito?

B.3.1 House Plan



ELENCO DISPOSITIVI SMART DELLA CASA

Autoparlante funzionalità di attivazione evento disponibili: 1. Rilevamento passaggio 2. Scoccare di un'ora 3. Comando vocale	Luce Armadio	Termostato
Lampadina	Tapparella	Macchina caffè
Sensore di movimento	Serratura intelligente	Lampada comodino