The Workshops of the Thirtieth AAAI Conference on Artificial Intelligence
Artificial Intelligence Applied to Assistive Technologies and Smart Environments:
Technical Report WS-16-01

An Intelligent Dialogue Agent for the IoT Home

Heesik Jeon, Hyung Rai Oh, Inchul Hwang, and Jihie Kim

Intelligence Solution Team, Samsung Electronics Co., Ltd., Seoul, Korea
{heesik.jeon, hyungrai.oh, inc.hwang, jihie.kim}@samsung.com

Abstract

In this paper, we propose an intelligent dialogue agent for the IoT home. The goal of the proposed system is to efficiently control IoT devices with natural spoken dialogue. This system is made up of the following components: Spoken Language Understanding for analyzing textual input and understanding user intention, Dialogue Management with a State Manager that consists of dialogue policies, Context Manager for understanding the environment, Action Planner responsible for generating a sequence of actions to achieve user intention, Things Manager for observing and controlling IoT devices, and Natural Language Generation that generates natural language from computer-based representation. This system is fully implemented in software and is evaluated in a real IoT home environment.

1 Introduction

An increasing number of devices around us nowadays are getting connected to the Internet. This goes much beyond being merely connected to the Internet, and we call this emerging global trend in IT the Internet of Things (IoT). Many IoT devices we use today are already connected to the Internet, and we see new devices being released in the market every day. We normally use apps, including an integrated service app, or smart assistant based on a dialogue system, to control IoT devices. In doing so, we either use separate or remote apps, or an integrated app. An alternative way to control IoT devices is to use the smart assistant, based on a dialogue system.

Let's assume that you just got home from work. Normally, you would conduct the following actions: turning on the light in the living room and the dress room, turning it off after getting changed. You will then turn on the light in the bathroom, take a bath and turn off the light. You will go to the living room, sit on your couch to watch a movie. Using a remote controller, you turn on the TV and set sound out on external speakers to get better sound quality than the one from a built-in speaker. You may feel that the light in the living room is too bright, so you get up, turn down the light and you can finally lounge on your couch and watch your movie. What if you can get all of these actions done simply by saying "I'm home", or "play movie"?

We have faced many technical challenges here. The system must be able to accurately spot the inconvenience faced by the user even without being notified each time, and also be aware of the extent of its capacity to resolve the issue. Having such ability would ultimately allow the system to be executed in a way that resolves the user inconvenience in the most effective fashion. In recognition of such need, we came up with an intelligent dialogue system to provide a human-centric interface.

Nowadays, a rule-driven autonomous agent is very common and is widely used in industries. In a real environment however, it faces limited scalability and flexibility, as well as constant conflict resolution. Thus, the proposed system includes a planner to generate relevant plans for achieving user intentions or actions with as little user interaction as possible, and also without any predefined rules. It is designed based on the Hierarchical Task Network (HTN) [Nau et al, 2005] planning technique with high level of context and knowledge base.

In session 2 of this paper, we explain recent approaches to an intelligent agent using dialogue system. We then propose an Intelligent Agent for the IoT Home in Session 3, followed by an explanation on the performance evaluation in Session 4.

2 Related Works

Many approaches can be taken toward an Intelligent Agent using dialogue system. The dialogue system can be seen in two ways: initiative and dialogue management.

Initiative management system operates in three ways-system-initiative, user-initiative and mixed-initiative. System-initiative mostly asks questions to the user to obtain necessary information, whereas user-initiative takes requests from the user and the user takes control of the conversation. Mixed-initiative has combined advantages of

both initiatives, and although the system is supposed to be in control of the conversation, it allows users flexibility to provide more information regarding user intention, or change the task/topic.

We have defined two categories within the dialogue management system. . The first takes approaches that are frame-based, form-based, task-based, plan-based, topic-based, goal-based and agenda-based. The second approaches are knowledge-based, data-driven and hybrid dialogue management. The knowledge-based uses finite-state automata which often involves handcrafted rules and has been deployed in many practical applications thanks to its simplicity. The data-driven dialogue management uses reinforcement learning based on Markov Decision Processes (MDPs) [Roy, et al, 2000] or partially observable MDP (POMDPs) [Williams, et al, 2008] to allow changes to the dialogue policies. The hybrid dialogue management [Henderson et al, 2005] integrates the two approaches to optimize dialogue policies.

Planning, as the name suggests, generates plans to achieve a specific goal and is the key component of an intelligent agent. During the last few years, the efficiency of planning has been greatly enhanced by many researchers. The planning techniques can be classified into four approaches. The first approach by GRAPHPLAN [Blum & Furst, 1997] described a new plan generation technique based on a planning graph. It significantly reduced the plan generation time at this time. The second approach, planning by the Satisfiability method [Kautz & Selman, 1996] translated a problem with planning into a propositional satisfiability problem, and was much faster than any other techniques. The third approach, a heuristic-search planning [Bonnet & Geffner, 1998] proposed a heuristic search algorithm from an initial state to a goal state with low computation complexity. However, these approaches still face limits in terms of computation complexity because they were developed based on trial-and-error search of a large space of possible solutions. The fourth approach, HTN, is a popular and widely employed planning technique which addressed a search-control strategy called ordered task decomposition over the task tree. Baier and McIlraith proposed HTNPlan-P [Baier and McIlraith, 2009] that generates preferred plans based on a HTN planner. GraphHTN [Lotem, et al, 1999] introduced a HTN planner with an applied hybrid planner and GRAPHPLAN to improve the performance.

3 An Intelligent Agent for the IoT Home

As shown in Figure 1, the proposed system is composed of the following components: SLU-inclusive Dialogue System, State Manager (ST), Context Manager (CM), Action Planner (AP), Things Manager (TM), Knowledge Base (KB) and Natural Language Generation (NLG).

There are many intelligent personal assistants available in the market today, such as Samsung S Voice, Google Now, Apple Siri and Microsoft Cortana. Among the existing systems are a user initiative system and a mixed initiative system. To run the user initiative system, the user has to make utterances such as “Hi Galaxy” or “OK Google”, press a button to obtain information, give a command or start a conversation. It means that the user takes control of the conversation and decides what to do with the system. With the mixed initiative system, both the user and the system can perform initiatives.

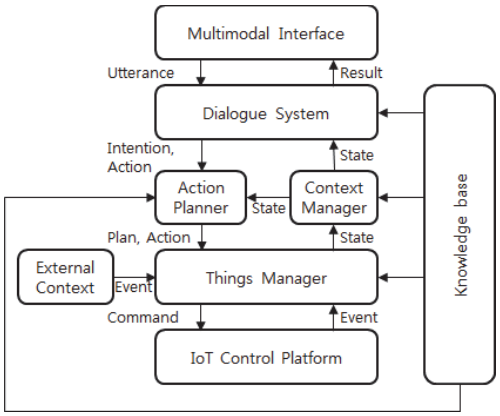

Figure 1 Overall architecture of Intelligent Dialogue Agent for the IoT Home

This paper proposes a mixed-initiative intelligent agent based on an agenda-based dialogue system with connecting KBs. The Agenda-based approach [Rudnicky et al. 1999] is an ordered list of topics, represented by handlers that govern certain single items or a collection of information. The system has connected knowledge bases (KBs) as well as state specifications that are similar to the finite-state and can be adjusted by using the incremental learning.

3.1 Spoken Language Understanding

Given that there are many ways to express a single meaning in natural language, the goal of SLU is to detect meaningful information such as a named entity or user intention, as shown in Table 1. To this end, we elaborate primitive actions as the main_goal: on/off, activate/deactivate, start/stop, connect/disconnect, up/down, next/previous, lock/unlock, open/close, play/pause/rew/ff, mute/unmute, set, select, and dim, to name a few. Furthermore, a single utterance can hold multiple intentions and also the negation.

A Hybrid Approach to User Intention Modeling for Dialogue Simulation [Jung et al. 2009] defines: userIntention = [dialog_act, main_goal, component_slot], where dialog_act is a domain-independent label of an utterance at

the level of illocutionary force (e.g. command, statement, question) and main_goal is the domain-specific user goal of an utterance. Component slots represent domain-specific named-entities in the utterance.

dialog_act	main_goal	components
question	wh_question	[O] object array consists of name / value, negation [O] attribute object array such as target, location, time, plural, negation
	yn_question	
statement	things	
	location	
	number	
	status	
command	on/off	
	start/stop	
	play/pause(hold)	
	open/close	
	…	
	rule	[M] conditions [M] tasks
answer	positive	[O] object including name / value
	negative	

Table 1 User intention

We use rules and machine learnings (MLs) to figure out user intentions. Rule-based classifiers with storage of 350 rules recognize user intentions from the patterns elaborated by experts, and it consists of rules to identify dialog_act, main_goal, and components (target, location etc.). ML-based classifiers use SVM, CRF, n-grams and W2V to recognize the same categories as the Rule-based module. Over 10,000 corpora of user utterances are trained and tested, using a cross-validation method. Its pipeline is shown below in Figure 2.

Figure 2 Natural Language Processing/Understanding pipeline

3.2 State Manager

The State Manager is a central flow controller. It has 4 main tasks: 1. Using SLU, extracting necessary information from user utterance and detecting user intention. 2. Storing extracted information in the short-term memory and using user intention as a source of "event" and "trigger" in the state specifications. The short-term memory remembers contexts, state, essential named entities, and previous conversations that took place during certain periods of time. It allows the system to handle multi-state seamlessly. 3. Using the user intention, deciding what to do according to the state specifications. The state specifications, Figure 3, are the main policies that can be adjusted by using the incremental learning. 4. Communicating with the internal/external modules and systems such as CM, Question Answering (QA), KB, AP, and NLG.

The state specification consists of an array of three objects: name, transitions and actions. "Name" is a name of the state. "Transitions" consists of an array of two objects, namely, "event" which refers to conditions for changing the state and "state", which is a name of the state. "Actions" has a sequence of actions that can run simultaneously and consists of an array of two objects, "trigger" and "action".

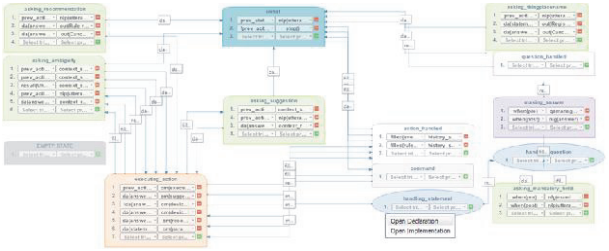

Figure 3 Examples of State specifications

3.3 Context Manager

The Context Manager provides important information to DM and AP. It has 4 main tasks: 1. Disambiguating action, target IoT device, and location. 2. Understanding high-level context via TM. 3. Notifying information and suggesting plans or actions by detecting user activities and circumstances. 4. Storing personalized information in the long-term memory.

The CM clarifies action, target device and location, while minimizing user inconvenience. For example, if there is only one light in the user's house, the system detects it and turns it on. If there are two lights in the house and one of them is already turned on, the system turns the other light on upon the user's command. When all the lights in the house are turned on, CM will send the information to SM by saying "All your lights are on". Also, when both lights in the house are turned off, the system asks the user which one the user wants to turn on. The user can give answers such as "all", "both", "living room", "bedroom", "forget it", "cancel" to clarify what he/she wants.

The CM includes a context reasoning mechanism to extract high-level context about user situation and device state. For example, CM estimates room temperature with primitive state of devices. It can simplify parameters of AP and helps its communication with human via DM.

The system can notify/make a suggestion to the user. For example, if the user is watching TV in the living room and the laundry is just done, the system can display a message "Laundry done" on the user's TV screen or on his/her mobile phone. The way system notifies the user can be adjusted according to the user's preference. The system can also suggest the user to close the window if the window is open when it rains.

The system stores user's name, schedule, preferences, and IoT devices including the setting and location in its long-term memory. This gives the system the ability to call the user by his/her name, and remind the user to take necessary actions, such as going out for lunch with friends, turning on the fan instead of the air conditioner by user preference, and turning off all devices in the living room.

3.4 Action Planning

The Action Planner is adapted to control IoT devices by analyzing natural language. The traditional planners are designed to solve complex problems with goal, initial state, operation, pre-condition, and post-condition with many parameters. However, DM can obtain intentions and a few entities such as product name, device type, label and some parameters with ambiguity. Thus, it is necessary to communicate between AP and KB and compensate missing parameters with as little user interaction as possible. HTN planner is one of the efficient solutions in this scenario, as it defines task and method that are well structured to integrate with KB. Figure 4 shows an example of a planning tree of a movie playing task; an example of user utterance would be "I want to watch Terminator". It includes another implicit task of making the best environment for watching the movie. Thus, a movie playing task has two parameters – X is a content description and Y is a value of light to control brightness. However, user's utterance doesn't have a specific value of light. The AP tries to obtain value from user preference knowledge or common sense knowledge, suggests dim-room task to the user (dash-line means recommendation) and chooses a method between light dim down and light off, considering capability of the light.

Plan Scoring: The AP employs a score function to find the best plan. We consider two score functions to see if 1. one device is better than the other to perform an action, and to see 2. which action is more appropriate to perform certain task in a device. The AP calculates the total score of a plan with two mixed score functions. We assume that N plans are generated by the AP and the total score of the i^{th} plan (s_i) can be calculated by

$$s_i = \sum_{j=1}^{L} r_k(A_j) \Big/ L, \quad i \in \mathbf{P}, k \in \mathbf{D}$$

Where L is the number of actions in the i^{th} plan and $r_k(A_j)$ is a reward of the j^{th} action in the k^{th} device for user's goal. It's defined in KB and automatically updated with user feedback. ***P*** and ***D*** are a set of plans and devices, respectively. Finally, AP chooses a plan has the best score.

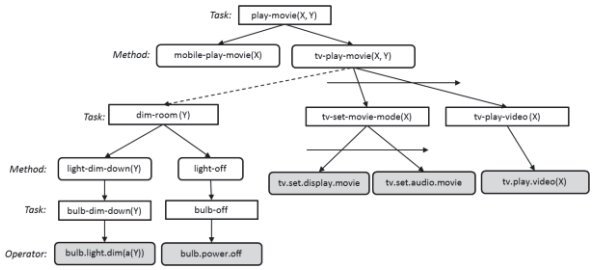

Figure 4 An example of a planning tree based on HTN. A dash line means an optional task

3.5 Things Manager

The Things Manager observes the state of IoT devices by receiving event from the IoT control platform and sends command back to the platform. It converts the plan received from the AP to a set of commands adapted to each IoT device. In addition, the TM handles additional information for reasoning, such as weather, as shown in Figure 1. In addition, it contains the location information of each device to support AP and CM.

3.6 Knowledge Base

The Knowledge Base stores usefully structured information for other components. There are relationships among action ontology, things ontology, common sense ontology, and user preference ontology, as shown in Figure 5.

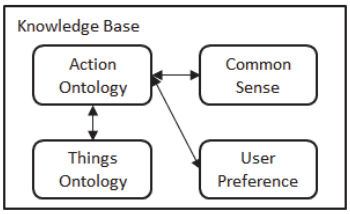

Figure 5 The relationship between ontologies in Knowledge Base

Things ontology contains information about home devices, such as their specifications, capabilities, and the methods to control them. It is an extension to capabilities, attributes and commands defined by SmartThings, which is an open platform for the IoT home services to connect, control and monitor IoT devices. Action ontology captures relations among task, sub-task, action, and condition based

on an HTN planner. We organized actions associated with home devices and related conditions. Each action has pre-conditions and post-conditions. Some actions require a certain state before being performed. We call this the pre-condition, while the state after performing the action is referred to as the post-condition. The information linked to individual devices at home, such as a cleaner, is defined in the thing ontology while user's personal preferences are defined in the preference ontology. The common sense ontology has in it the principles that govern the user behavior, such as reasons and motives behind different actions, general behavioral tendencies, circumstantial rules, requirements etc. The knowledge base is implemented in the form of OWL ontology [W3C OWL Working Group, 2009]

3.7 Natural Language Generation

The proposed Natural Language Generation generates natural sentences for answer, completion, confirmation, asks missing slots and ambiguities, and makes suggestions/recommendations based on rules and templates.

Types	NLG Output examples
Answer	The temperature is 25 degrees
Completion	Laundry done.
Confirmation	Would you like to turn on the light?
Missing slots Ambiguity	Which device do you want to turn on?
	Which light do you want to turn on, in the living room or bedroom?
Suggestion	It's raining. You'd better close the window.
Recommendation	You can automatically dim the light in the living room when you watch a movie. Would you like to register the automatic task?

Table 2 Examples of NLG result

The input to NLG contains the type of action defined in Table 2 and the list of objects such as verb, target, location and objects. Verb is the name of an action we defined in Table 1. Target is usually a device, location a place, object a feature of the target. We predefined a list of structures that have subject, object, verb particles etc.

4 Implementation and Evaluation

The proposed system is fully implemented in software and is evaluated in a real IoT home environment. For now, we covered the weather and 9 types of IoT devices such as TV (Samsung Tizen Smart TV), washer (Samsung Bubble), air conditioner (Samsung Q9000), air cleaner (Samsung AX9000), robot cleaner (Samsung POWERbot), thermostat (Honeywell), light (Philips Hue), speaker (Samsung 360), and sensors (Smart Thing), along with 167 actions linked to these devices.

We set up testbed rooms: a living room, master bedroom, front door and a laundry room. Each room has a speaker as an input device. IoT Things present in each of these rooms are shown in Table 3.

Place	IoT Devices
Front door	door sensor
Living room	TV, light(dimmable), air conditioner, robot cleaner, sensor(window)
Bedroom	light, air purifier
Laundry	light, washer
External Context	weather

Table 3 IoT Devices as components of the testbed rooms

The participants were informed of what IoT devices the testbed rooms have and what their capabilities are. The most frequent cases were single-intentions like "turn on the light", "turn off the TV in 30 minutes", "what's the weather like today?" and some cases were multi-intentions like "turn on the TV and off the light", as shown in Table 4. The utterance ratio (command: 79.52%, statement: 10%, question: 7.15%, and answer: 3.33%) was determined by analyzing utterances made by random groups of participants.

Turn on
Turn off the TV in 30 minutes
Clean home
Don't dim the light
Do I need to take my umbrella?
Turn on the TV and light
Turn on the TV and off the lights
Turn on the TV in the bedroom and off the light in kitchen
Turn off all lights except in the living room

Table 4 Examples of participants' utterances

Table 5 shows generated plans by AP. We could observe that AP chooses an external speaker as it offers better sound quality than the TV's internal speaker. AP selects dim light action, although the light has a turn off function. Finally, by suggesting closing a window to the user, AP makes an inference from the common sense ontology in KB.

Task	Examples of planning
Play movie	turn on TV → choose video content → set display mode → turn on external speaker → connect TV to speaker → set sound mode → dim light → play video
Make it cool	turn on air conditioner → set cooling mode → close the window if it is opened (suggestion)

Table 5 Examples of the generated plan by AP

There are two types of user commands: 'Action' is when a user gives a direct command, and 'Task' refers to a combination of different types of actions. Table 6 shows the performance of each module, according to the given command type. Here, SLU is observed to require longer elapse time than AP in the case of action, as it needs to extract entities such as target and object. AP on the other hand showed high level of computational complexity, as it needs to set up and execute a plan from the tasks. DM demonstrated same performance level under both commands.

Type	Average Elapse Time (ms)		
	SLU	DM	AP
Action	40	16	23
Task	28	16	175

Table 6 Average elapse time of main components. 'Turn on TV' and 'Play movie' are the sample utterances of action and task.

We conducted the evaluation, using 100 essential commands that cover most actions and 2200 extended utterances such as questions, commands including ambiguous cases, statements and answers that are unknown to engineers and are subject to constant changes. The estimated accuracy of user intention with extended utterances was 99.52% for dialog_act, 98.81% for main_goal, and 90.24% for components.

5 Conclusion

In this paper, we proposed an intelligent dialogue agent for the IoT home with SLU, for analyzing textual input and understanding user intention that are the keys to ask for information, give a command or start a conversation. SM-inclusive DM with policies that we introduced decides what to do according to the state specifications that can be adjusted by using the incremental learning and store extracted information in the short-term memory. CM's role is to understand the environment, notify information, suggest relevant plan or action, and store personalized information in the long-term memory to enhance user convenience. AP generates a sequence of actions to achieve user intention. TM observes and controls IoT devices, and NLG generates natural language from computer-based representation, based on rules and templates. Using the system above offers enhanced user convenience and puts the user in control of the IoT devices, as well as allowing them to plan actions at home.

For the possible future task, we would like to design our IoT devices recognize the user by the voice, as well as provide more information regarding dining, transportation, health care etc., hence further enhance the user convenience beyond the limits of space and time.

References

Baier, S. S. J. A., & McIlraith, S. A. (2009). HTN planning with preferences. In 21st Int. Joint Conf. on Artificial Intelligence (pp. 1790-1797).

Blum, A. L., & Furst, M. L. (1997). Fast planning through planning graph analysis. Artificial intelligence, 90(1), 281-300.

Bonnet, B., & Geffner, H. (1998). HSP: Heuristic search planner. In AIPS-98 Planning Competition Pittsburgh, PA.

Chung, G. (2004, July). Developing a flexible spoken dialog system using simulation. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (p. 63). Association for Computational Linguistics.

Henderson, J., Lemon, O., & Georgila, K. (2005, August). Hybrid reinforcement/supervised learning for dialogue policies from communicator data. In IJCAI workshop on knowledge and reasoning in practical dialogue systems (pp. 68-75).

Jung, S., Lee, C., Kim, K., & Lee, G. G. (2009, August). Hybrid approach to user intention modeling for dialog simulation. In Proceedings of the ACL-IJCNLP 2009 Conference Short Papers (pp. 17-20). Association for Computational Linguistics.

Kautz, H., & Selman, B. (1996, August). Pushing the envelope: Planning, propositional logic, and stochastic search. In Proceedings of the National Conference on Artificial Intelligence (pp. 1194-1201).

Kuter, U., Sirin, E., Parsia, B., Nau, D., & Hendler, J. (2005). Information gathering during planning for web service composition. Web semantics: science, services and agents on the World Wide Web, 3(2), 183-205.

Lee, C., Jung, S., Kim, K., Lee, D., & Lee, G. G. (2010). Recent approaches to dialog management for spoken dialog systems. Journal of Computing Science and Engineering, 4(1), 1-22.

Lotem, A., Nau, D. S., & Hendler, J. A. (1999, July). Using planning graphs for solving HTN planning problems. In AAAI/IAAI (pp. 534-540).

Nau, D., Au, T. C., Ilghami, O., Kuter, U., Wu, D., Yaman, F., ... & Murdock, J. W. (2005). Applications of SHOP and SHOP2. Intelligent Systems, IEEE, 20(2), 34-41.

Roy, N., Pineau, J., & Thrun, S. (2000, October). Spoken dialogue management using probabilistic reasoning. In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics (pp. 93-100). Association for Computational Linguistics.

Rudnicky, A., & Xu, W. (1999, December). An agenda-based dialog management architecture for spoken language systems. In IEEE Automatic Speech Recognition and Understanding Workshop (pp. 1-337).

W3C OWL Working Group. 2009. OWL 2 Web Ontology Language: Document Overview. Available at http://www.w3.org/TR/owl2-overview/

Williams, J. D., & Young, S. (2007). Partially observable Markov decision processes for spoken dialog systems. Computer Speech & Language, 21(2), 393-422.

Williams, J. D., Poupart, P., & Young, S. (2008). Partially observable Markov decision processes with continuous observations for dialogue management. In Recent Trends in Discourse and Dialogue (pp. 191-217). Springer Netherlands.