Making the Intelligent Home Smart Through Touch-Control Trigger-Action Programming

Guan Wang(✉) and Michael L. Littman

Brown University, Providence, RI 02912, USA
{wang,mlittman}@cs.brown.edu

Abstract. We introduces a new UI model of Trigger-Action Programming (TAP), that allows users to program through touch-control interfaces to create complicated tasks easily. We present three different user interfaces (UI), for each user interface, we analyze its advantages, limitations, and potential utility. We explain how our UIs can mitigate the problems of TAP caused by ambiguity and demonstrate why our UIs will benefit people without programming backgrounds.

Keywords: Trigger-action programming · Touch-control interface · Smart home · End-user programming

1 Introduction

Trigger-action programming (TAP) makes it possible for people with non-technical backgrounds to write programs in a high-level way. Existing tools, such as IF-THIS-THEN-THAT (IFTTT), enable users to control real devices and virtual services in an intelligent way by creating online tasks remotely. The user interface of IFTTT builds a "rule" as a one-dimensional "IF-A-Then-B" structure. To create a TAP rule, a user chooses from the list of applications supported by IFTTT and decides what A and B should be. This design enables autonomy in devices and services. An example rule is "IFit rains tomorrowTHENsend me an email". However, many useful tasks in real life have a higher-level of complexity that cannot be represented by the simple logic of "IF-A-THEN-B".[1]

On the other hand, research has shown that there are sometimes differences between a user's real intent, his semantic expression, and the program created by the user [2]. To mitigate this problem, which is caused by ambiguity, misunderstanding, and oversimplification of TAP systems, we introduce our own UIs that uses touch control to help end-users build TAP programs of higher complexity in an easy and accurate way.

2 Related Work

Our research builds upon previous investigations of TAP for context-aware systems, and more general end-user programming. Dey et al. discovered through

E. Sucar et al. (Eds.): AFI 2016, LNICST 179, pp. 75–78, 2017.
DOI: 10.1007/978-3-319-49622-1_9

their visual programming system iCAP that users often specified actions in a TAP-like manner when they are asked to think of behaviors for context-aware applications. [3] Pane et al. also stated similar conclusion that consistency exists between users' mental models and the IF-THEN style specification. [4] Other researchers have also discovered obstacles in applying TAP in smart homes. Newman et al. presented that designing rules which perform actions based on future states is often difficult, especially for none-programming-background users. [5] Ur et al. discussed the problem of users creating rules that require multiple events and conditions to occur at once, and their study showed that there are certain amount of goals in "smart homes" can not be done by the simple IF-A-THEN-B structure. [1] Conducting two studies to examine user's interpretations of TAP rules, Huang et al. claimed that there could be a significant discrepancy between a rule expressed in an IF-THEN style and the performance expected by the user who created the rule. [2] Other studies have also inspired our work, Ghiani et al. mentioned in their paper that lacking feedback could be a hindrance in TAP systems, and that contributed our idea of the "rehearse" function.

3 UIs

There are two purposes of our design: Enabling complex tasks to be constructed easily; Mitigating the problems caused by ambiguity. Given the trend toward mobile devices, we design our UIs based on touch control, making it easy and comfortable for users to create and edit tasks with smart phones or tablets. The design of our touch-control technology, "drag-and-link", is inspired by previous work such as the research project "Math Tutor" [6]. For each interface, users can use their fingers to click, drag, and link between objects in a two-dimensional space. The system checks the validity of the current program automatically, and will display a "REHEARSE" button in the top-right corner whenever a task is completed with no error. By clicking the button, the user will see a rehearsal of the program that he has just made, allowing him to tell whether the performance matches his interpretation.

Figure 1 shows the user experience of 3 UI. In UI 1, the interface is an unconstrained two-dimensional space. The search bar at the top allows the user to query applications. The bottom bar contains two rows of buttons, where the first row has "IF, THEN, WHENEVER, AS LONG AS, WHILE, AFTER", and the second row is "AND, OR, NOT". By adding these functional words, we enhance the original IFTTT statement model to allow the expression of higher-level tasks. The user can click anywhere on the screen to get a menu of icons of supported applications and services. After all icons are placed, the user can use his finger to link between them. If the link is valid, it will be shown as a purple arrow starting from the origin icon to the target icon. Whenever a syntactically valid task is detected by the system, the rehearse button appears.

UI 2 is an updated model of UI 1. We noticed that time is one of the most important features in TAP. Adding time features to each trigger and action

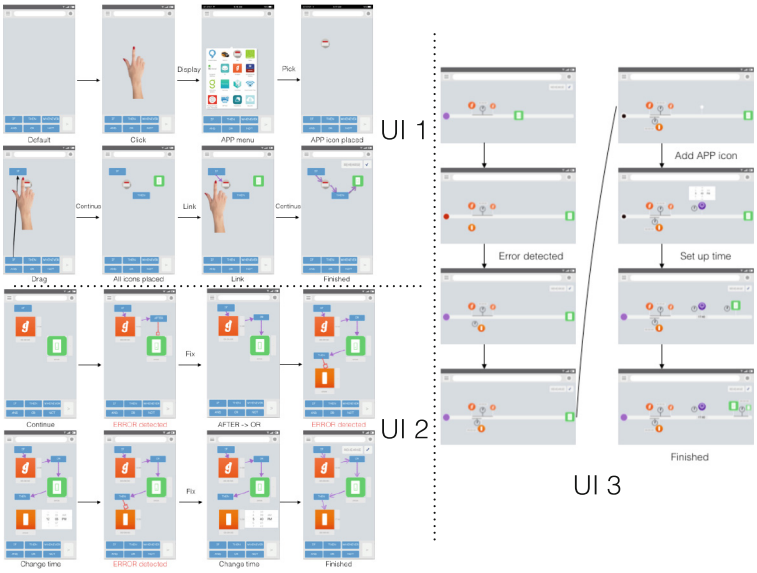

Fig. 1. User experience flow of UI 1, UI 2, and UI 3 (Color figure online)

reveals: (1) When a TAP ends; (2) The duration of a TAP; (3) Inherent conflicts like "IF-It's 6 pm-THEN-email me at 5 pm"; (4) Possible ambiguities by helping the user understand the time sequence better. UI 2 can detect logical errors that are syntactically valid, but do not make sense in real life. When the user wants to add an application icon, he will need to set up two features: The duration is represented as a rectangle attached to the bottom of the app icon; The time that the app starts is activated as a rectangle attached to the right of the app icon. The default setting of duration is "instantaneously", which means that if no duration number is given, the app will run and only run once instantaneously. Similarly, the rule will be activated ever since it's checked in by default.

UI 3 comes with a unique feature, a timeline, which shows the time sequence directly and obviously. By clicking at the screen, a timeline will appear, which allows the user to drop pins (APP icons) onto it. For each pin, the user can slide it to the right to activate the feature of duration. Once the system detects a valid TAP, a rehearse button will appear similar to the behavior of other 2 UIs.

4 Discussion

Our UIs mitigate the problems of ambiguity via using time features, and higher-level statements. The original IFTTT expression cannot deal with situations like: (1) what to do when a TAP ends; (2) how long each trigger or action should last; (3) how many times will a TAP execute; and (4) recipes with more than one trigger. In our UIs, we use "AND, OR, NOT" to represent the logic functionality. We also introduce "AFTER, WHILE, AS LONG AS, WHENEVER"

to enhance the triggering features. With the help of these functions, the user can create complex tasks in a much easier and simpler way. Our UIs decrease the distance between semantic expression and non-technical background users. Moreover, in the classical model of IFTTT, some TAPs are syntactically valid but can never happen in real life. It's hard for users to notice these types of problems in TAP, and sometimes it's not easy for users to even understand what has gone wrong. With the help of time-sequence features and warning signals, the problematic recipe components are displayed straightforwardly. Users can now learn "what's wrong" immediately and fix the error by identifying and changing the erroneously component accurately. Last, but not least, the rehearse function guarantees the TAP result can be shown to the users before checked in.

On the other side, our interfaces require more work to create a TAP. Users may need to spend more time figuring out how functions work, which could lead to a higher chance of making mistakes. Our UIs can be applied to web-apps and universal apps, which allow users to set up tasks to control intelligent devices at home in a smart way. Anything that is supported by the current IFTTT system will be supported by our UI models as well. Moreover, devices and services connected through Smartthings hub can also be covered and controlled by our system. With the nature of its high-level statements, error detection, rehearse functionality, our UIs can be a useful tool to introduce classical programming.

Acknowledgement. Members of the Brown UPOD Research Group provided valuable feedback. Stephen Brawner came up with suggestions. Blase Ur, Tushar Bhargava, Jiyun Lee, Diane Schulze, and Jared Cohen also gave helpful feedback.

References

1. Ur, B., McManus, E., Pak Yong Ho, M., Littman, M.L.: Practical trigger-action programming in the smart home. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 803–812. ACM, April 2014
2. Huang, J., Cakmak, M.: Supporting mental model accuracy in trigger-action programming. In: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing, pp. 215–225. ACM, September 2015
3. Dey, A.K., Sohn, T., Streng, S., Kodama, J.: iCAP: interactive prototyping of context-aware applications. In: Fishkin, K.P., Schiele, B., Nixon, P., Quigley, A. (eds.) Pervasive 2006. LNCS, vol. 3968, pp. 254–271. Springer, Heidelberg (2006). doi:10.1007/11748625_16
4. Pane, J.F., Myers, B.A.: Studying the language and structure in non-programmers' solutions to programming problems. Int. J. Hum. Comput. Stud. **54**(2), 237–264 (2001)
5. Newman, M.W., Elliott, A., Smith, T.F.: Providing an integrated user experience of networked media, devices, and services through end-user composition. In: Indulska, J., Patterson, D.J., Rodden, T., Ott, M. (eds.) Pervasive 2008. LNCS, vol. 5013, pp. 213–227. Springer, Heidelberg (2008). doi:10.1007/978-3-540-79576-6_13
6. Wang, G., Bowditch, N., Kwon, M., Zeleznik, R., Laviola, J.: A tablet-based math tutor for beginning algebra. In: 2015 Workshop on the Impact of Pen and Touch Technology on Education (WIPTTE 2015), Redmond, WA, 28-30 April (2015)