Using Contextual Knowledge to Resume Human-Agent Conversations when Programming the Intelligence of Smart Environments

Asterios Leonidis[1], Margherita Antona[1] and Constantine Stephanidis[1,2]

[1]Foundation for Research and Technology – Hellas (FORTH) – Institute of Computer Science (ICS) (GREECE)
[2] University of Crete, Department of Computer Science (GREECE)
{leonidis, antona, cs}@ics.forth.gr

Abstract. This paper presents a hybrid technical solution towards addressing conversational interruptions when interacting (via a typed interface) with a virtual agent to program the intelligence of a smart environment. The AmI Solertis system is a ubiquitous programming environment that facilitates the definition of the behavior of a Smart Environment. To address the issue above, Ami Solertis introduces a mechanism that stores any unexpectedly interrupted conversations in a stack along with relevant contextual information. This context-sensitive information attached to the dialog, is used to re-establish a detailed context in the user's working memory when resuming human-agent conversations, within.

Keywords: Ambient Intelligence, Conversational Agent, Smart Environment Programming.

1 Introduction

A shift is currently perceived from the "one person with one computer" paradigm, which is based on explicit human computer interaction, towards a ubiquitous and pervasive computing landscape, where implicit interaction and continuous cooperation is becoming the norm of computer supported activities. This modern way of living, where technology and information are flowing around the physical environment, led to the emergence of the Ambient Technology paradigm [4], where physical objects are enhanced with computer technology to communicate, share information and collaborate with other technological devices in an intelligent fashion. AmI is a prominent dimension in ICT, while industrial stakeholders have already acknowledged its benefits and opportunities and introduce to the mass market digital devices and services that will transform traditional environments to technologically enhanced "intelligent" ones [13].

In order to maximize the efficiency, extensibility and adaptation to the needs of their users, AmI systems need to be programmable. In fact, their programmers are not expected to be only professional developers, but also inexperienced end-users who can either modify the behavior of the system based on their current needs or extend its intelligence even further to address future necessities. The latter, in combination with the

fact that programing such environments is inherently difficult due to their high architectural and computational complexity, further complicates the overall process. To that end, the Ambient Intelligence Research Programmed of ICS-FORTH has developed AmI Solertis [11], a studio for Ambient Intelligence applications development, which empowers users to create behaviors scenarios by reviewing and modifying the high-level "business logic" of a smart environment in a user-friendly manner through both a visual programming platform and an accompanying chat-bot agent. This paper aims to demonstrate the technical solution applied by the AmI Solertis system in order to handle interruptions that occur while a human actor is engaged in a conversation with a chat-bot agent trying to define the parameters and deploy a new script that dictates the intelligent behavior of the Smart Environment. AmI Solertis resides on contextual knowledge to smoothly resume the conversation by both adapting the virtual agent's behavior and bridging the human counterpart's mental gap due to the context-switch.

2 Related Work

The emerging paradigm of end-user programming along with the rapid development of the Internet-of-Things and of Smart Environments strongly suggest that in the near future the end-users will have to be able to modify the behavior of the software artefacts they possess [7, 12]. To that end, various alternatives have been proposed, with visual programming paradigm being the prominent choice since it facilitates inexperienced users to quickly learn how to build simple programs [5]. Nevertheless, the rise of chat-bots [15] and the fusion of conversational interfaces into Smart Environments enables the provision of more intuitive interaction paradigms (i.e., natural language dialogues) between the user and the intelligent virtual agents (i.e., technological artefacts).

Apart from their undeniable benefits though, various challenges surface including the necessity to handle the interruptions that might occur. A great amount of research [14] has been conducted on how to handle interruptions that occur during human-agent conversations and mainly originate from the human participant. However, in the context of this work, the term conversational interruption refers to the unexpected termination of the user's interaction with one of the intelligent agents' due to external stimuli. The employed handling strategy is analogous to the approach for addressing the problem of task switching from the Human-Computer Interaction (HCI) perspective (e.g., breadcrumbs) [9] and the established guidelines on how to design mobile experiences for partial attention and interruption (e.g., data segmentation, glanceability) [6].

The key aspect is the provision of relevant visual information (i.e., context) to the user to assist his wayfinding. In the case of AmI Solertis where conversation dialogues are used to build structured programs (i.e., behavior scripts), the availability of relevant visual information is rather limited. Therefore, this work synthesizes a hybrid approach; on the one hand, it introduces custom conversational models relying on utterances modelling and annotation to understand natural language [17] and on hierarchical dialog models to implement the conversational frameworks [2, 18] that structure and assess the progress of the expected interaction, and on the other hand it attaches appropriate contextual knowledge to act as mental cues that facilitate conversation resume.

3 Modelling Framework

This work has been applied in the context of AmI Solertis system, which facilitates management of the available technological artifacts of a Smart Environment by its occupants. For that to be achieved, AmI Solertis offers, amongst others, a virtual agent in the form of a chat-bot, named ADAM (Ambient and Distributed Agents' Manager), that can communicate with the end-users via natural language textual interface in order to help them accomplish numerous orchestration-related tasks (i.e., inquiries about services' status, definition of new behaviors, validation of existing ones, etc.). As expected, within such environments interruptions and task-switching are quite frequent, therefore ADAM incorporates functionality that enables recovery from such situations either by resuming them immediately in case the interrupt ends soon, or by providing relevant contextual information to assist the user recall the initial objective at a later time. A collection of meta-models that store domain-specific conversation information has been designed and used by ADAM's Conversational Interruption Handler (CIH).

Modelling Conversations. As aforementioned, in the context of this work, certain types of conversational dialogs exist that are directly mapped to the available back-end facilities of the AmI Solertis framework, namely: (i) Activation and Deactivation Requests, (ii) Exploration Inquiries, (iii) Monitoring Inquiries, (iv) Recommendation Inquiries, (v) Creation and Modification Commands and (v) Help and Training Dialogs.
Given that ADAM supports certain user tasks (e.g., define a new behavior), every dialog corresponds to a micro finite-state machine (FSM) [16] and populates the respective data structures as the dialog with the user progresses and the FSM transits from one state to the next. An illustrative example of such a state machine is presented in Fig. 1; the model of "Create a new behavior" is decomposed into its inner states and the three alternative dialogues are provided to demonstrate ADAM's in-order (dialogue 3) and out-of-order (dialogues 1 and 2) data collection.
In addition to the FSM-specific meta-model that stores the behavior related parameters, the ADAM's Dialog Manager (DM), which orchestrates the overall process, stores any active conversations with their closure [10]. In particular, it persists the entire environment of a conversation including: (i) its complete stack of utterances of the current session, (ii) the identified user intents (i.e., what is the task that the user aims to accomplish via the current dialog), and (iii) any entities that have been successfully recognized and will be forwarded to the services of the AmI Solertis framework to execute the actual task (e.g., instantiate and deploy a new script that encodes the desired behavior).

Modelling Contextual Knowledge and Interruptions. Context of Use is the cornerstone of Smart Environments as it constitutes the glue that brings together all the isolated services under a common roof and enables the environment's proactive response to user needs by permitting intelligence sharing across the various services (including virtual agents). In such environments, various sensors and applications collect and process huge amount of information to distil and distribute useful insights in the form of small information chunks (e.g., users' location, activities at hand, state of interactive

applications, etc.) that can be persisted and used by other agents. Such abundant contextual information is the key towards semantically annotating interruptions, as it contains background data that will be used to provide mental cues to the user and assist conversation continuation.

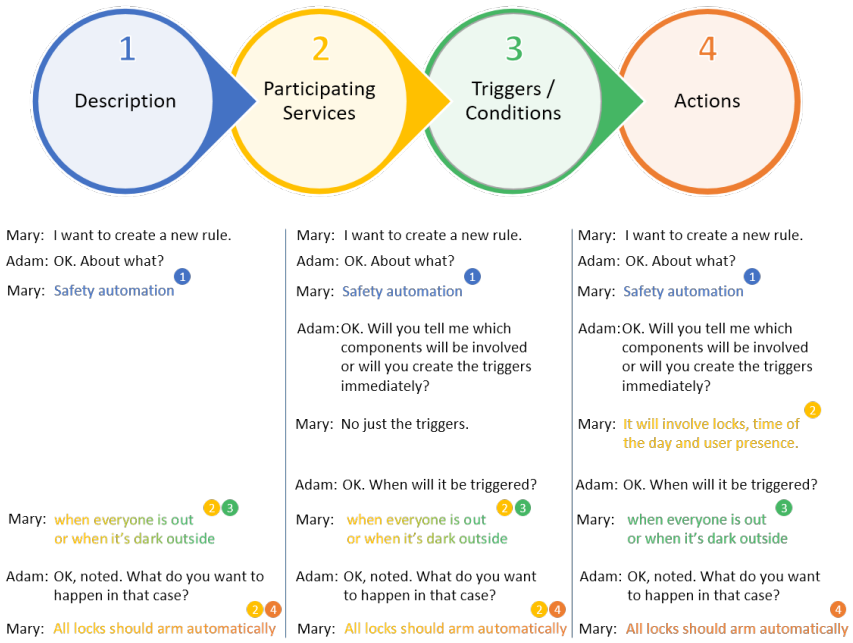

Fig. 1. The state machine of the "**Create a new rule"** dialog and the out-of-order population of the mandatory fields of the relevant AmI Solertis command.

As an example, consider the following scenario: Mary, while watching TV, notices the trailer of a new TV show that is about to air. The show seems interesting, so she decides to ask ADAM to program its recording at the relevant time. At that moment, her friend Anna has decided to pay her a visit and is standing at the front door. As expected, the latter event is of higher priority, therefore interaction with ADAM gets aborted. The dialog has not finished yet, however contextual information has been recorded (Fig. 2); in particular, the Dialog Manager has stored that: (i) Mary has started telling ADAM to create a new behavior, (ii) she was watching TV, (iii) the TV was on channel X, (iv) the program airing at that moment was Show-Z and (v) the trailer is about Show-W. In the future, when Mary is available, ADAM can make a suggestion to resume their conversation and, if necessary, assist her by recalling what she was about to ask by retrieving and presenting that information.

Apparently, the amount of information available at any given moment in a fully connected smart environment is quite large; nevertheless, appropriate filtering techniques

are used to determine which contextual information is useful to be attached to the interruption event (e.g., presence of another user, activities/applications capturing the user focus, etc.). The AmI Solertis framework handles information storage using its internal logging mechanisms and exposes to ADAM's Conversational Interruptions Handler only their MIME-types [3] and their reference points that facilitate their retrieval. Since ADAM's behavior is encoded using the AmI Solertis facilities, it can be easily extended via plugins (i.e., behavior scripts) to introduce new strategies that rely on contextual information that were not originally available; e.g., in the context of the aforementioned scenario, if a TV channel can be queried to preview its content at that time through a 10-seconds video (including any commercials shown), then ADAM could orchestrate its presentation to further assist Mary.

Fig. 2. Filtered contextual information is attached to an interruption event.

4 Resuming conversations: the AmI Solertis approach

Modelling by itself can be easily used by ADAM's submodules to handle a single conversation that has been interrupted; but interruptions happen all the time in our daily environments and in many cases, they happen concurrently. The objective of employing ambient intelligence technologies is not just to limit unwanted interruptions, but mainly to minimize their handling and recovery time [8] benefitting from contextual information and collective intelligence.

4.1 Stack-based interruption handling

The approach followed by the AmI Solertis framework regarding conversational interruption handling is inspired by the methodology followed by almost all compilers, to manage their run-time memory as a stack [1]; whenever a procedure is called, space for

its local variables is pushed onto a stack, and when the procedure terminates, that space is popped off the stack. In more details, whenever a new conversation starts, ADAM's Dialog Manager creates a new activation record about that dialog, where the relevant references to the appropriate models are stored (Fig. 3). In case of an interrupt, the current dialog's contextual information is persisted, the overall dialog is marked as incomplete, and it gets pushed to the stack of dialogs for future reference. Additionally, while the user is occupied handling the unexpected event, ADAM's Conversational Interruptions Handler attempt to evaluate the importance and the priority of the latest conversation and estimate the duration of the interruption, in order to determine whether an intervention for resume should be scheduled as soon as the user is available. For instance, if Mary has started saying "Tonight, schedule lights to…" or "When my child approaches…" and the interrupt originated from an expected event which is not supposed to take long (e.g., a courier delivering a package), then ADAM will mark that last conversation as critical and will prompt Mary to resume at the earliest opportunity. In different cases, other appropriate closure strategies will be applied.

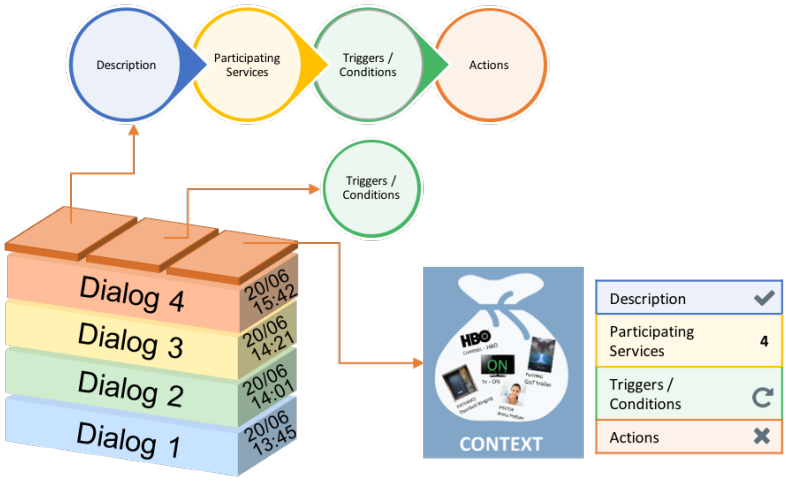

Fig. 3. Snapshot of the Stack of Incomplete Conversations and a decomposed activation record.

4.2 Strategies for Reaching Conversational Closures

Proactive Informative Interruptions. A conversation model, in the context of this work, refers to a single or a series of commands that will be submitted to the AmI Solertis framework for execution. Every command specifies a set of mandatory fields (i.e., intents, entities) that must be populated using information extracted from the user's input. Therefore, if ADAM identifies that those values are not properly set before submission, it can proactively interrupt the conversational, by altering its natural flow, in order to get additional clarifications (e.g., user-specific jargon has been identified in a mandatory filed instead of the actual service's name).

Interruption Avoidance or Fast-track. ADAM, apart from artificially injecting interruptions in active conversations, aims to predict: (a) whether interruptions are about to happen and (b) how much time will their handling require, in order to determine how to avoid them. In case of a potential conflict, ADAM either suggests to postpone the currently active conversation; e.g., if the user is about to get a notification saying that the cooking process needs his attention -a task that is expected to take 2 minutes to be completed-, whereas rule creation process takes 5 minutes, then ADAM will ask for user's permission to schedule it for another time, or fast-track interruptions (e.g., asks the user to check the pot now rather than waiting for two minutes first) to ensure that the conversation will not stop from that point forward.

Conversation Abandonment. ADAM periodically monitors the context of use in order to determine which pending conversations need to be discarded. To that end, ADAM examines the participating services, triggers and actions of every partial activation record and evaluates whether an equivalent behavior has been already installed in AmI Solertis from any other configuration channel (e.g., manually via the supported graphical editor [11], automatically by the AmI Solertis self-assessment mechanism, etc.).

Conversation Continuation. Most importantly though, ADAM's Interruption Handler aims to successfully complete any interrupted conversation. To that end, it monitors interaction and if a similar dialog begins or an analogous context is detected, it suggests to resume a relevant pending conversation. Priority is given to conversations that were recently interrupted (e.g., a few seconds ago), then to incomplete conversations that can be finished quickly (e.g., only the confirmation step is missing) and finally, to conversations that were classified as important or time-critical by the relevant AmI Solertis facilities (e.g., involve sensitive family members, the user implied that the relevant behavior should start today, etc.). On resume, if the conversation was conducted some time ago, ADAM provides a short summary to the user (e.g., "You were saying about Safety Automation") to improve recollection. If necessary, ADAM can re-estate a more detailed context of use in user's working memory by restoring the information attached to that dialog; e.g., "You were watching a commercial about Show-Z on Channel X, when you asked me to create a new rule to… Maybe you wanted to set a recurring notification to remind you about that show or record it if you are unavailable?", or "You were saying that when the night falls, for you family's safety, you want to lock…", etc.

5 Conclusions and Future Work

This paper presented a hybrid approach towards addressing conversational interruptions when interacting with a virtual agent to program the intelligence of a smart environment. The AmI Solertis approach integrates existing HCI strategies to address task switching with modelling of conversational dialogs and introduces the notion of runtime interruption handling as a stack, using contextual knowledge to resume human-agent conversations. The added value of employing ambient intelligence technologies is not just to limit unwanted interruptions, but mainly to minimize their handling and

recovery time, by re-establishing a more detailed context of use in the user's working memory using the information attached to the dialog at hand.

To investigate the performance of this approach in its actual target environment and whether it fulfils its goals, a user-based study is planned to take place in the near future in a smart home setup located in the Ambient Facility building at the premises of the Institute of Computer Science (ICS) of the Foundation for Research and Technology (FORTH) in Heraklion, Crete.

References

1. Ahom, A.V. et al.: Compilers, Principles, Techniques. Addison Wesley Boston (1986).
2. Bellegarda, J.R.: Spoken Language Understanding for Natural Interaction: The Siri Experience. In: Natural Interaction with Robots, Knowbots and Smartphones. pp. 3–14 Springer, New York, NY (2014).
3. Borenstein, N.S., Freed, N.: Multipurpose internet mail extensions (MIME) part two: Media types. (1996).
4. Cook, D.J. et al.: Ambient intelligence: Technologies, applications, and opportunities. Pervasive Mob. Comput. 5, 4, 277–298 (2009).
5. Green, T.R.G., Petre, M.: Usability Analysis of Visual Programming Environments: A "Cognitive Dimensions" Framework. J. Vis. Lang. Comput. 7, 2, 131–174 (1996).
6. Hinman, R.: The mobile frontier. O'Reilly Media, Inc. (2012).
7. Holloway, S., Julien, C.: The Case for End-user Programming of Ubiquitous Computing Environments. In: Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research. pp. 167–172 ACM, New York, NY, USA (2010).
8. Kanner, A.D. et al.: Comparison of two modes of stress measurement: Daily hassles and uplifts versus major life events. J. Behav. Med. 4, 1, 1–39 (1981).
9. Krug, S.: Don't make me think!: a common sense approach to Web usability. Pearson Education India (2000).
10. Landin, P.J.: The mechanical evaluation of expressions. Comput. J. 6, 4, 308–320 (1964).
11. Leonidis, A. et al.: Enabling Programmability of Smart Learning Environments by Teachers. In: Distributed, Ambient, and Pervasive Interactions. pp. 62–73 Springer, Cham (2015).
12. Lieberman, H. et al.: End-User Development: An Emerging Paradigm. In: Lieberman, H. et al. (eds.) End User Development. pp. 1–8 Springer Netherlands (2006).
13. McKinsey and Company: There's No Place Like [A CONNECTED] Home, http://www.mckinsey.com/connectedhome/.
14. Nakano, Mikio, et al. "A two-layer model for behavior and dialogue planning in conversational service robots." Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on. IEEE, 2005.
15. Shawar, B.A., Atwell, E.: Chatbots: are they really useful? In: LDV Forum. pp. 29–49 (2007).
16. Sklyarov, V.: Hierarchical finite-state machines and their use for digital control. IEEE Trans. Very Large Scale Integr. VLSI Syst. 7, 2, 222–228 (1999).
17. Stolcke, A. et al.: Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. Comput. Linguist. 26, 3, 339–373 (2000).
18. Williams, J.D. et al.: Fast and easy language understanding for dialog systems with Microsoft Language Understanding Intelligent Service (LUIS). In: 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue. p. 159 (2015).