# End User Personalization of Social Humanoid Robots

Fabio Paternò, Marco Manca, Carmen Santoro

CNR-ISTI, HIIS Laboratory
Via Moruzzi 1, 56124 Pisa, Italy
{fabio.paterno, marco.manca, carmen.santoro}@isti.cnr.it

**Abstract.** In this position paper we present some research challenges for end user personalization of social humanoid robots. We introduce the motivations for addressing such challenges and the main features of the type of robots that we want to consider. We discuss some initial research efforts that have recently been put forward in this area, and the type of solutions that have been proposed in order to facilitate the development activities for people without programming experience. We then identify and discuss some research challenges that can be important to address in the near future in order to better exploit such emerging technologies.

**Keywords:** End User Development, Social Humanoid Robots, Personalization, Internet of Things.

## 1 Introduction

Robots are increasingly used in many contexts. According to Gartner, by 2025, three out of 10 jobs will be converted to software, robots or smart machines[1]. The term "robot" is rather broad, since there are many types of robots. We can distinguish between industrial robots and social humanoid robots. In the former case they are robots that accomplish specific tasks in specific work contexts with the goal to perform repetitive and well-defined activities (such as to pick an object and place it in a given location) in a more efficient way with respect to humans.
Social humanoid robots are different since they can interact with us by voice, gestures and all the other modes typical of human communication. They can help us in housework, care of children, elderly and disabled people, hospitals, hotels … Since the behaviour of such robots can react to many types of events and involve the performance of various types of actions, which depend on the specific context of use in which they are deployed, their complete behaviour cannot be hardcoded at design time by developers who cannot foresee all the possible situations that they can encounter during their use.



[1] https://www.gartner.com/binaries/content/assets/events/keywords/symposium/sym25/gartner-sym24-executive-report2.pdf

Thus, a key challenge for obtaining satisfying robot applications is the customization of their behaviour to meet the specific, and often evolving users' needs. In this perspective, a solution with a potential high impact is to have everyday users to directly specify the robot behaviour they need.

End-User Development (EUD) [10] is a research field that is stimulating increasing interest, and aims to support non-professional developers to create or modify their applications. Over time this area has evolved in order to address new emerging technological trends. It started with the graphical desktop systems, next it moved to consider the possibility of Web applications with their open interfaces, then it considered the mobile devices as a platform for the relevant activities, recently its adoption in Internet of Things (IoT) contexts has been addressed [3, 4, 8]. We think that a new fundamental challenge in this area is to investigate how it can be supported with social humanoid robots.

## 2 Possible Approaches

All the available robots can be programmed through some language, usually oriented to engineers. The issue of making the development of robot applications easier has started to be considered [1, 2, 6]. The Pepper robot by Softbanck Robotics can be programmed through Choreographe [11], which supports an iconic data flow visual language. Each icon corresponds to a Python procedure. In simple cases the composition of such modules works well and can be efficient. However, it requires the ability to understand the underlying Python language to actually be able to apply it. In case of several icons, it does not scale well since the many associated connecting arrows can soon result in a complex representation rather difficult to interpret. The use of some block-based programming languages [12] has recently been proposed for example by Weintrop [14]. They propose a block-based interface (CoBlox) for programming a one-armed industrial robot. The results show that participants using the block-based interface successfully implemented programs faster with no loss in accuracy while reporting higher scores for usability, learnability, and overall satisfaction. However, they considered a rather limited scenario, with one-armed industrial robots, and the participants were asked to program a "pick and place" routine, which is a relatively narrow set of functionality. Such solutions seem to work well when they consider scenarios in which the possible options to address are limited. However, modern humanoid robots can flexibly react to many possible events and perform a wide variety of actions. In addition, it would be interesting to have approaches integrating the control of the robot behaviour with what happens in the surrounding environment.

A possible approach to addressing such issues is trigger-action programming. Its main features are a compact and intuitive structure connecting dynamic situations with expected reactions, and it does not require the use of complex programming structures. Thus, it can be suitable for people without programming experience who have not algorithmic abilities. This type of approach so far has mainly been used for automating Web services or home control. The most common tool in this area is IFTTT [9, 13], which allows users to create rules where triggers and actions can be chosen by existing

services (e.g. Facebook, Weather, Dropbox, etc.). Thus, users can create rules such as "when someone tags me on Facebook then send me an email". In this perspective, it can be interesting to investigate whether a trigger-action paradigm can enable people without particular programming knowledge to personalize the behaviour of humanoid robots. In addition, new services can be created through the combination of the monitoring capabilities associated with IoT and the robots' capability of acting and interacting with the environment and especially humans.

With this approach personalization rules have the format: when something happens (trigger) an action should happen (action) in the format IF/WHEN <triggers> DO <action>. A trigger can be composed of event(s) and/or condition(s). Events are changes of some aspect that occur at a given time in the application or in some contextual aspect (they can be indicated with the WHEN keyword). Conditions are specific states that are verified for some time in some aspects of the context or application state (they can be indicated with the IF keyword). Actions are the effects and can be performed in various ways by the robot and/or the surrounding devices. In this way, it is possible to specify rules such as "When I look like sad do the robot tells a joke" or "When someone arrives at the door do turn on the light and the robot asks *who are you* ?"

Recently, a solution following this approach has been presented [5]. It is based on a personalization platform composed of various modules: a personalization rule editor, a rule manager, and a context manager. When the rules are created they are sent to the rule manager, which subscribes to the context manager in order to be informed when the relevant triggers occur. The rule manager is composed of a server and various context delegates associated with the various sensing technologies used. One of such context delegate has been implemented in Python for the Pepper robot so that it can communicate with the low-level robot sensors, and then send relevant information to the context server. When the rules are triggered the associated actions are sent to the robot and/or the surrounding devices for execution. A first user test has been carried out in which users had to: indicate rules that would consider useful with the robot; specify and execute with the rule editor three rules with increasing complexity; specify and execute with the rule editor three predefined rules that were provided with natural language description. The users were able to perform such tasks without performing significant errors. The time of performance was short. Only when they specified the first rule it took longer with respect to the other cases because they had to familiarise with the tool. The qualitative comments were positive, the only issues were with the voice recognition that in a few cases did not work well immediately, and the robot execution of the actions, which in a few cases was slower than expected.

The results of the user study indicate that trigger-action rules can be actually used for personalizing the robot behaviour and is well received by users. However, the study was an in-lab test, in which explicit task assignments were given in order to limit the possibility of ambiguity and to better compare the collected results. Thus, new studies are necessary to challenge users through less explicit task instructions, and conduct longitudinal studies, assessing the use of the tailoring tool for longer periods of time, and investigating whether further aspects emerge (e.g. if and how the way to personalise the robot would change over time).

## 3 Discussion

If we consider the reported experiences and approaches we can identify various research issues that need to be addressed in the near future.

One general issue is the choice of the most relevant metaphor and programming style. Various proposals have been put forward in EUD [10], and a definitive answer for the most suitable for social humanoid robot has not yet been given. One further aspect is that EUD approaches should offer the most appropriate abstraction level for people without programming experience, so that they can identify the relevant triggers without having to learn the low-level sensing technology and communication protocols involved, and set the robot reactions that are expected to be similar to those of a human by using a non-technical vocabulary.

The EUD solutions should provide effective support to specify human-like robot behaviour, which is more complex than that of various appliances. For example, to control a light it is sufficient to send a command that specifies light level, colour, and duration, while a robot behaviour may require to specify gestures, words to pronounce, movements. Thus relevant approaches should be able to support constructs able to handle the specification of multiple actions with different durations. One aspect that can open up the possibility to create various innovative services is the support of the integration of the humanoid robot behaviour with what happens in the surrounding environment that can be detected through various types of sensors and smart objects.

When adopting an approach based on rules it can be useful to provide the possibility of simulating the execution of the rules and support conflict analysis. The simulation would allow people to check that the rules really perform the desired behaviour, which is not always clearly understood. For example, the difference of events and conditions is something that requires some attention in order to be correctly applied. In addition, when several rules are created it may happen that some of them require conflicting actions for example at a given time one rule asks for moving the robot arm up and another one down. One possible approach to end user debugging of personalization rules is reported in [7].

Graceful degradation of robot behaviour is a feature that can be positively appreciated by end users. It means that the robot is able to provide them with relevant explanations when its abilities for some reason degrade, thus trying to mitigate associated negative effects. Lastly, support of a high degree of personalization rules reuse is certainly useful. Thus, relevant EUD environment should support such features along with the possibility to share rules amongst users.

## 4 Conclusions

Our life will soon become an interactive experience with various types of robots, smart objects and devices. Professional developers cannot predict the specific needs of each user, it is thus important to provide novel environments for end user personalization of the available applications and technologies. In this position paper some interesting research challenges from this perspective are discussed.

## References


1. S. Alexandrova, M. Cakmak, K. Hsiao, and L. Takayama. 2014. Robot Programming by Demonstration with Interactive Action Visualizations. In Proceedings of the 2014 Robotics: Science and Systems Conference. DOI: https://doi.org/10.15607/RSS.2014.X.048
2. N. Buchina, S. Kamel and E. I. Barakova. 2016. Design and evaluation of an end-user friendly tool for robot programming. In Proceedings of IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN '16). IEEE, 185-191. DOI: https://doi.org/10.1109/ROMAN.2016.7745109
3. G. Desolda, C. Ardito, M Matera, Empowering end users to customize their smart environments: model, composition paradigms, and domain-specific tools, ACM Transactions on Computer-Human Interaction (TOCHI) 24 (2), 12
4. Ghiani, G., Manca, M., Paternò, F., Santoro, C.: Personalization of context-dependent applications through trigger-action rules. ACM Trans. Comput.-Hum. Interact. 24(2), Article No. 14 (2017)
5. N. Leonardi, M. Manca, F. Paternò, C. Santoro, Trigger-Action Programming for Personalising Humanoid Robot Behaviour. ACM Conference on Human Factors in Computing Systems (CHI'19), Glasgow, Paper 445.
6. J. Huang and M. Cakmak. 2017. Code3: A system for end to-end programming of mobile manipulator robots for novices and experts. In Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI '17). ACM, New York, NY, USA, 453-462. DOI: https://doi.org/10.1145/2909824.3020215
7. M. Manca, F.Paternò, C. Santoro, L Corcella, Supporting end-user debugging of trigger-action rules for IoT applications, International Journal of Human-Computer Studies, Vol.123, 56-69
8. P Markopoulos, J Nichols, F Paternò and V Pipek (2017). End-User Development for the Internet of Things, ACM Transactions on Computer-Human Interaction (TOCHI) Volume 24 Issue 2, 9, May 2017
9. X. Mi, F. Qian, Y. Zhang, X. Wang: An empirical characterization of IFTTT: ecosystem, usage, and performance. IMC 2017: 398-404
10. F Paternò, C Santoro, A design space for end user development in the time of the internet of things, in New Perspectives in End-User Development, 2017, Springer Verlag, pp.43-59
11. E. Pot, J. Monceaux, R. Gelin, B. Maisonnier. 2009. Choregraphe: a graphical tool for humanoid robot programming. In Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN'09). IEEE, 46-51. DOI: https://doi.org/10.1109/ROMAN.2009.5326209
12. M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, and Y. Kafai. 2009. Scratch: programming for all. Commun. ACM 52, 11 (November 2009), 60-67. DOI: https://doi.org/10.1145/1592761.1592779
13. B. Ur, M. P. Y. Ho, S. Brawner, J. Lee, S. Mennicken, N. Picard, D. Schulze, M. L. Littman: Trigger-Action Programming in the Wild: An Analysis of 200, 000 IFTTT Recipes. CHI 2016: 3227-3231
14. D. Weintrop, A. Afzal, J. Salac, P. Francis, B. Li, D. C. Shepherd, and D. Franklin. 2018. Evaluating CoBlox: A Comparative Study of Robotics Programming Environments for Adult Novices. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18). ACM, New York, NY, USA, Paper 366, 12 pages.