Challenges in Multi-User Interaction with a Social Humanoid Robot Pepper

Peter Forbrig

University of Rostock, Department of Computer Science Chair of Software Engineering, Albert-Einstein-Str. 22, 18055 Rostock, Germany
peter.forbrig@uni-rostock.de

Abstract. The paper discusses the challenges of engineering applications for humanoid robots like Pepper. Such robots can be used in different domains. We focus on the training aspect for patients after a stroke. The humanoid plays the role of a trainer. It should motivate the patient but also pushes when necessary. It should recognize the need for brakes and provide social interactions. A specific challenge is the collaboration of a patient, a supporting person and the robot. This collaboration should be modeled before implementation. Additionally, a model of the three roles is necessary. Based on the analyzed situation an appropriate interaction has to be selected. Domain specific languages might be helpful for modeling and engineering the applications.

Keywords: User model · Cognitive model · Activity models · Domain specific language.

1 Introduction

Robots are used in different domains. They are used to support industrial production [14], give advice or support healthcare [1]. Some robots are totally functional like in production lines for cars others look like animals and are used as substitute for pats. Some robots look like humans and are classified as humanoids. Pepper is one of this kind of robots. It is a social humanoid robot that is able to recognize faces and basic human emotions. Pepper is optimized for human interaction with voice and gestures. The robot is able to engage with people through conversation and a touch screen [9]. Fig. 1 gives an impression how Pepper looks like. It looks cute.

Fig. 1. Humanoid robot Pepper

Pepper can be programmed in Java with the QiSDK plug-in for Android Studio. Fig. 2 provides a screenshot of the programming environment with the plugin extensions for the robot. The plugins allow the connection with a real or a virtual robot. Additionally, on can see the virtual simulator of Pepper. It looks like the real robot and has arms and fingers but wheels instead of legs.

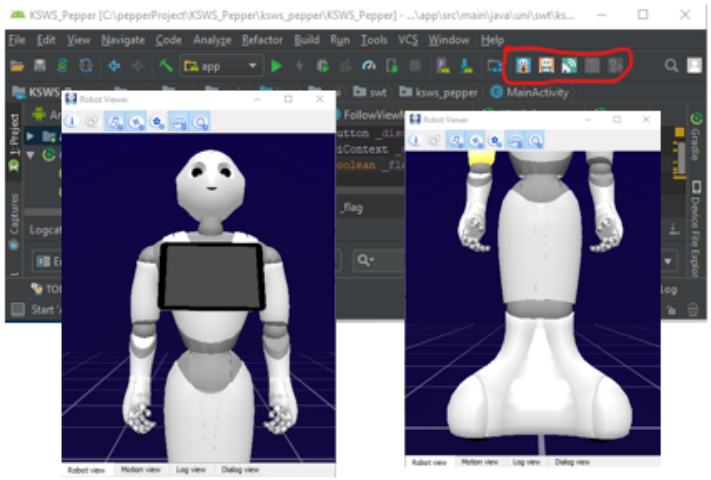

Fig. 2. Android Studio with Pepper plugin and simulator of the humanoid robot Pepper

In our project we want to study whether robots like Pepper are able to motivate patients for their exercises. Currently, exercises are observed by humans and can be provided only two or three times per week. However, it would be much better if patients would be able to train several times per day.

Unfortunately, success results of the training come very slowly. Even that there is some success after a week, patients often do not recognize it. That is the reason, why they often question the sense of their exercises. Therefore, motivation is very important. We assume that we can support the motivation of a patient with the help of the robot Pepper. One option is the expression of the observed success by Pepper in an appropriate way.

Together with psychologists, sociologists and experts from medicine we want to study which kind of patients can be supported by Pepper in which way. We want to provide answers for questions like: What kind of interaction is appropriate in certain circumstances? When does a patient need a support? When does it make sense to suggest a break? How can motivation be increased? What kind of models can be helpful? How can we engineer applications for Pepper in an appropriate way?

2 Exercises for Patients after Strokes

Our project was initiated by Professor Thomas Platz who has been working on Neurohabilitation for several years. He invented some specific exercises for arms that stimulate the brain of stroke patients. An evaluation of his approach is published in [13]. Fig. 3 gives an impression of some of those training activities.

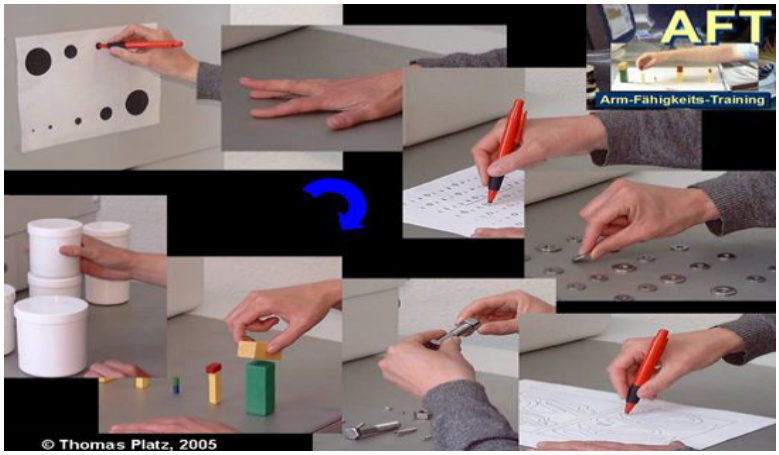

Fig. 3. Stimulating the brain by arm ability training

In the left upper corner of Fig. 3 one can see how a patient has to hit different circles. The time for this exercise is two minutes. A patient has to hit the two largest circles first, then the two smaller ones and so on. After hitting the smallest one the patient has to start from the beginning again. Currently, a physiotherapist counts the correct hits. If this training is supported by a tablet implementation one can count the hits and their accuracy more precisely. The robot Pepper can comment on the reached results and motivate a patient with sentences like: Great, you reached a new level! Let us try again, we nearly reached the best result! Today, you are really in a good mood. Let us make a break and try to reach the same result again!

Some patients have at the beginning such problems with their arm that they are not able to hold it by themselves. They need a supporter that provides guidance by holding the elbow of the patient in its hand. This can be boring for the supporter after a while. Therefore, both the patient and the supporter need some motivation for the training. Pepper has to recognize if the motivation of one of the collaborators drops. If the motivation level of the patient and the supporter falls at the same time it has to be decided with whom to interact first. This decision has to be made based on the expected cognitive state of the collaborators. Knowledge representations have to be specified for such application.

Research has been focused during the last years on the interaction of a computer with one user. Two users and a greater variety of interaction modes provide new challenges. There are only few papers about Robot interaction with several people. Tahir et al. [15] discuss an experiment where a humanoid robot acts as a mediator between two persons. Glas et al. [10] report about a system called Interaction Composer. The system is "a visual programming environment designed to enable programmers and non-programmers to collaboratively design social human-robot interactions in the form of state-based flows." [10] However, they report success and failure.

As a first attempt we will try to use our language DSL-CoTaL [7]. It is text based and allows the generation of models for CTTE [12], HAMSTERS [2] and CoTaSE [3].

```
role pepper {
    root roo1 = greet >> instruct >> observe{*} [> stop_observing >> bye
        task stop_observing pre patient.allInstances.finishes_exercises
        task instruct pre patient.allInstances.greet

        task observe = congratulate []
                       congratulate_enthusiastic []
                       support []
                       provide_strong_support >>
                       save_data
        task congratulate pre patient.oneInstance.perform_correct
        task congratulate_enthusiastic pre patient.oneInstance.perform_several_correct
        task support pre patient.oneInstance.perform_wrong
        task provide_strong_support pre patient.oneInstance.perform_several_wrong

            task congratulate = happy1[nice]
            task congratulate_enthusiastic = happy3[excellent]
            task support = support1[shoulder, try_again]
            task provide_strong_support = support2[right_arm, make_break]
}
```

Fig. 4. Task model for Pepper in DSL-CoTaL[7]

Fig. 4 provides an impression how the language looks like. The behavior of Pepper is specified as a task model in a rule-based way. A task on the left hand side of a rule is specified by sub-tasks on the right side. Some tasks need a precondition before they can be executed. Preconditions are expressed in an OCL-like[6] way. The task *stop_observing* e.g. can be executed when all instances in the role of *patient* have finished their exercises. Pepper starts his task *root1* with greeting that is followed by providing instructions. Afterwards, several observations are possible. The iteration is stopped when *stop_observing* is performed. Finally, Peppers says good bye. Generic components like discussed in [8] are used to specify *congratulate*, *congratulate-enthusiastic*, *support* and *provide_strong_support*.

The task model of the patient and of the supporter are omitted here. They do not provide new insights. However, a simple collaboration model can be seen in Fig. 5. It is written in the style of the collaboration model of CTTE [12]. A training session consists of three sequential parts. First, the robot , the patient

and the supporter greet each other in parallel. Second, Pepper provides some instructions and afterwards the patient performs her or his exercises. Third, all collaborators say good bye.

```
team coop {
    root training = greeting >> train >> finish
       task greeting = pepper.greet |||
                       patient.greet |||
                       supporter.greet
       task train =  pepper.instruct >>
                     patient.perfoms_exercises{*} [>
                     patient.finishes_exercises
       task finish = pepper.bye |||
                     patient.bye |||
                     supporter.bye
}
```

Fig. 5. Model of collaboration

Fig. 6 demonstrates possible definitions of different versions of happy.

```
component happy1[pa] {
    root happy1<pa> = raise_left_arm ||| say_<pa>
}

component happy2[pa] {
    root happy2<pa> = raise_both_arms |||
                      say_<pa> |||
                      turn_around
}
component happy3[pa] {
    root happy3<pa> = raise_both_arms |||
                      say_<pa> |||
                      turn_around |||
                      blink_with_eyes
}
```

Fig. 6. Generic Components similar to [8]

Especially for the activities of the robot Pepper it makes sense to specify a new domain-specific language (DSL) that allows the code generation for the Android Studio. Tasks like *raise_left_arm* would become keywords in that language.

3 Research Questions

In conjunction with the project E-BRAiN we identified the following research questions:

- How can applications with social robots like Pepper be engineered?
- What kind of interaction is appropriate for which kind of patient or supporter in which context?
- How can a concept be developed for systems with two persona types in connection with two user models? Which ideas can be reused from two-level personas [4]?
- How can the idea of under specification [5] be used for applications of Pepper?
- How has the DSL-CoTaL to be extended to allow appropriate collaborate task specifications?
- Which DSL allows best the specification of the behavior of the robot Pepper in such a way that the model can be animated and additionally, allows code generation for Android Studio?
- Which software architecture is appropriate for adaptive interactive applications for Pepper and further devices [11]?

These questions (and definitely some more) will guide us in our project during the next three years.

4 Summary and Outlook

The challenges in engineering of social humanoid robots like pepper were discussed. In context of the project E-BRAiN collaborative systems involving a robot, a patient and a supporter have to be developed. Managing the appropriate interaction technology (voice, gesture, lights, tablet, other devices) is a challenge. Domain-specific languages seem to be one way to specify necessary models. Some ideas were discussed and some research questions were presented. Further ideas are expected from the discussion during the workshop.

References

1. Ahn, H.S., Choi, J., Moon, H., Lim, Y.: Social human-robot interaction of human-care service robots. In: Companion of the 2018 ACM/IEEE International Conference on Human-Robot Interaction. pp. 385–386. HRI '18, ACM, New York, NY, USA (2018). https://doi.org/10.1145/3173386.3173565, http://doi.acm.org/10.1145/3173386.3173565
2. Bernhaupt, R., Palanque, P.A., Drouet, D., Martinie, C.: Enriching task models with usability and user experience evaluation data. In: Bogdan, C., Kuusinen, K., Lárusdóttir, M.K., Palanque, P.A., Winckler, M. (eds.) Human-Centered Software Engineering - 7th IFIP WG 13.2 International Working Conference, HCSE 2018, Sophia Antipolis, France, September 3-5, 2018, Revised Selected Papers. Lecture Notes in Computer Science, vol. 11262, pp. 146–163. Springer (2018). https://doi.org/10.1007/978-3-030-05909-5_9, https://doi.org/10.1007/978-3-030-05909-5_9

3. Buchholz, G., Forbrig, P.: Extended features of task models for specifying cooperative activities. PACMHCI **1**(EICS), 7:1–7:21 (2017). https://doi.org/10.1145/3095809, https://doi.org/10.1145/3095809
4. Dittmar, A., Hensch, M.: Two-level personas for nested design spaces. In: Proc. of CHI '15. pp. 3265–3274 (2015)
5. Dittmar, A., Kühn, M., Forbrig, P.: A domain-specific model-based design approach for end-user developers. In: Paternò, F., Santoro, C., Ziegler, J. (eds.) ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS'14, Rome, Italy, June 17-20, 2014. pp. 161–166. ACM (2014). https://doi.org/10.1145/2607023.2610275, https://doi.org/10.1145/2607023.2610275
6. Flake, S.: Towards the completion of the formal semantics of ocl 2.0. In: Proceedings of the 27th Australasian Conference on Computer Science - Volume 26. pp. 73–82. ACSC '04, Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2004), http://dl.acm.org/citation.cfm?id=979922.979932
7. Forbrig, P., Dittmar, A., Kühn, M.: A textual domain specific language for task models: Generating code for cotal, ctte, and HAMSTERS. In: Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2018, Paris, France, June 19-22, 2018. pp. 5:1–5:6. ACM (2018). https://doi.org/10.1145/3220134.3225217, https://doi.org/10.1145/3220134.3225217
8. Forbrig, P., Martinie, C., Palanque, P.A., Winckler, M., Fahssi, R.: Rapid task-models development using sub-models, sub-routines and generic components. In: Sauer, S., Bogdan, C., Forbrig, P., Bernhaupt, R., Winckler, M. (eds.) Human-Centered Software Engineering - 5th IFIP WG 13.2 International Conference, HCSE 2014, Paderborn, Germany, September 16-18, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8742, pp. 144–163. Springer (2014). https://doi.org/10.1007/978-3-662-44811-3_9, https://doi.org/10.1007/978-3-662-44811-3_9
9. Gardecki, A., Podpora, M.: Experience from the operation of the pepper humanoid robots. In: 2017 Progress in Applied Electrical Engineering (PAEE). pp. 1–6 (June 2017). https://doi.org/10.1109/PAEE.2017.8008994
10. Glas, D.F., Kanda, T., Ishiguro, H.: Human-robot interaction design using interaction composer: Eight years of lessons learned. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction. pp. 303–310. HRI '16, IEEE Press, Piscataway, NJ, USA (2016), http://dl.acm.org/citation.cfm?id=2906831.2906884
11. Herdin, C., Märtin, C., Forbrig, P.: Sitadapt: An architecture for situation-aware runtime adaptation of interactive systems. In: Kurosu, M. (ed.) Human-Computer Interaction. User Interface Design, Development and Multimodality - 19th International Conference, HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings, Part I. Lecture Notes in Computer Science, vol. 10271, pp. 447–455. Springer (2017). https://doi.org/10.1007/978-3-319-58071-5_33, https://doi.org/10.1007/978-3-319-58071-5_33
12. Mori, G., Paternò, F., Santoro, C.: CTTE: support for developing and analyzing task models for interactive system design. IEEE Trans. Software Eng. **28**(8), 797–813 (2002). https://doi.org/10.1109/TSE.2002.1027801, https://doi.org/10.1109/TSE.2002.1027801

13. Platz, T., Roschka, S., Doppl, K., Roth, C., Lotze, M., Sack, A.T., Rothwell, J.C.: Prolonged motor skill learning a combined behavioural training and theta burst tms study. Restorative Neurology and Neuroscience **30**(3), 213–224 (2012)
14. Seo, S.H., Gu, J., Jeong, S., Griffin, K., Young, J.E., Bunt, A., Prentice, S.: Women and men collaborating with robots on assembly lines: Designing a novel evaluation scenario for collocated human-robot teamwork. In: Proceedings of the 3rd International Conference on Human-Agent Interaction. pp. 3–9. HAI '15, ACM, New York, NY, USA (2015). https://doi.org/10.1145/2814940.2814948, http://doi.acm.org/10.1145/2814940.2814948
15. Tahir, Y., Rasheed, U., Dauwels, S., Dauwels, J.: Perception of humanoid social mediator in two-person dialogs. In: Proceedings of the 2014 ACM/IEEE International Conference on Human-robot Interaction. pp. 300–301. HRI '14, ACM, New York, NY, USA (2014). https://doi.org/10.1145/2559636.2559831, http://doi.acm.org/10.1145/2559636.2559831