User Interface Design for Internet of Things and Intelligent Agents Systems

Geert de Haan

Wittenborg University of Applied Sciences
Apeldoorn, the Netherlands
geert.dehaan@wittenborg.eu, geert.de.haan@upcmail.nl

ABSTRACT
This paper discusses a research and development position with respect to the human-robot interaction, in the areas of interacting with intelligent assistants, agent-based systems and Internet of Things (IoT) applications and interactive environments. Starting with a brief overview of the research behind the position, we list a number of lesson-learned from the projects concerned, and move on the stress the need for a user-centered, lightweight and flexible approach to the design of human-interface of intelligent systems as a combination of user-centred design (UCD) with co-design and co-creation methods.

Keywords
position paper: design method, user-centred design, agile design, agent-based systems, intelligent systems, ubiquitous computing, Internet of Things.

INTRODUCTION
Design in Human-Computer Interaction (HCI) of more in general Human-technology Interaction has developed in the course of time, determined by the state of the hardware and software technology, and the resulting affordances for HCI on the one hand, and the demands from the context in which the HCI systems are used, on the other hand. In the current ICT technology era, traditionally distinct IT functions such as data collection, processing and data access have converged communications into small, mobile, and networked devices, which provide functions or services that are no longer tied to a specific time or (work) place. Ubiquitous computing shows how ICT can penetrate our entire habitat, yet disappear as a visible technique (Weisser, 1991). In addition, telephone hardware and new media (software) applications allowed us to communicate and access information resources almost anytime and everywhere.

INTELLIGENT SYSTEMS DESIGN INSIGHTS
The background of this paper lays in research into human interaction with intelligent assistants (de Haan, 2000; de Haan et al., 2005), designing intelligent interfaces to agent-based systems (de Haan, 2003), and work on design methods for IoT and ubiquitous systems (de Haan, 2015). This research is not about ‘robots’ in the strict and humanoid sense of the word but each of these systems may be implemented within a robot-like interface, either as an interactive voice, a talking head on a screen, or as intelligent behaviour hidden inside a search-engine or even an interactive environment. In all of these cases, the essential part is the communication and interaction between human beings and some intelligent agent. As an example, in the Comris project (de Haan, 2000), the key design question was how to enable an intelligent assistant or agent to inform the user without disrupting the present activities of the user; hence: how to talk to the user without getting her out of his or her real world context. Some of the lessons-learned from this research track can be summarised as follows:

- don’t bother the user with everything that may be worthwhile of telling, provide a selection.
- text messaging is far less intruding within the user’s context then spoken messages are
- in designing intelligent systems, it is essential to adapt the ‘sophistication’ of the interaction to the particular type of user
- during the design process, allow for design exploration and co-design within the (real-world) context of use

SOFTWARE DESIGN FOR NEW MEDIA
In comparison to the design of software systems for business processes or pay rolling, the design process of new media products like interactive websites and mobile apps is lightweight, where flexibility with respect to adapting to changes in the market or the customers' wishes is a key requirement to the design and the design process. Because of the flexibility and ease of changing and updating media products, the design process is very lightweight and it uses a variety of informal tools without much reliance on the design notation (de Haan, 2015).

The design process of media products is based on prototypes, ranging from low-fidelity prototypes including paper prototypes, mock-ups and sketches to increasingly higher fidelity prototypes including clickable prototypes and the design product itself. Secondly, the media design process is a features-driven process, where each design cycle (or Scrum sprint; Schwaber and Beedle, 2002) focuses on the next most important features to implement. Finally, the media design process is an incremental design process with iteration both during the design process, as well as iteration after the design process, since maintenance is regarded as including further adaptation of functionality and presentation to evolving user wishes and tastes.

Media products tend to allow for flexible design methods because of the distinction between the 'front-end', the website or user interface of the system and the 'back-end', the database(s) that contains all collected sensory data and collection of links to other data and services used in the application. The strict separation of the user interface and the data processing part of the application allows for easy adaptation of the front-end whilst keeping the backend stable. Consider a website as an example. While a website is up and running, it is possible to present different groups of users with a different front-end, depending for example, on the basis of the local webserver they use. Next, data collected online about user preferences, conversion rate or sales figures may be used to choose the most successful front-end design. Naturally, such a process of online optimization is not restricted to a single trial but may take the form of a continuous process of adapting the looks and behaviour of a website or mobile app to the behaviour of its users.

New media design is also flexible because of the so-called “mashup” software-architecture: media applications follow a client-server architecture where a central but lightweight program script derives most of its functionality from calling external servers to provide the data from databases, sensor information from sensor networks, location information from location and map servers, etc.

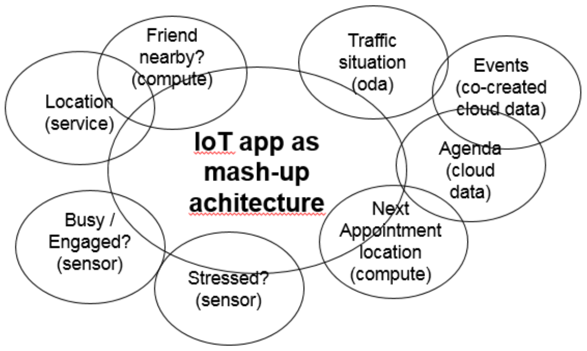

Figure 1. The Mashup software architecture

The mashup architecture makes it very easy to add, change or replace functions and service providers. If a particular sensor turns out as less useful, or a location service is too expensive or too cumbersome, you simple change to a different sensor or plug in another location service. In a typical IoT application, sensors produce data that is transmitted wirelessly (using 3G, WiFi or Low Power Wan) to an internet gateway. Once data is available at the gateway, it may be treated as any other data source, and put in a database, processed and made available by a server, like any other service.

Finally, because of the separate front-end and back-end, in combination with the mashup architecture, there is no inherent need for a complete or consistent design or software specifications a particular time. Features, functions and subroutines may be specified, designed and added when suitable without regard to particular design stages. As such, media application design may be feature-driven like in e.g. Scrum (Schwaber and Beedle, 2002) in a piece-by-piece fashion: incrementally and iteratively. As such, the design and the implementation process may proceed almost completely independent, allowing for a genuine user-centred design process featuring exploration, user participation, co-creation and co-design, paper- and rapid prototyping, etc. (van Dijk et al., 2011; Sanders and Stappers, 2008).

INTERFACING INTELLIGENT SYSTEMS

In this section, we will describe how, in the projects mentioned before, the design of the user interface or human interface, took place. In all projects (cf. de Haan, 2000; de Haan et al., 2005, de Haan, 2003; and de Haan, 2015) the principle aims was not to design a humanly usable system but rather to design a sound technical system in order to prove the feasibility. As an example, in the I-Mass project (de Haan, 2003) the aim was to design a system that is able to use a variety of different pre-given and language-specific databases to answer user questions in the area of cultural heritage. For example: what is clair-obscure, who invented it and give some me some examples. As an agent-based system, the answers would be found by creating a bunch of agents to search the databases using the language specific translation of the term, and selecting the best candidate answers and presenting them in the users native language and level of expertise, possible in different modalities such as text, verbal messages, etc.

In this project, to design the user interface, we used two techniques. First, we used a co-design technique (avant la lettre), in which cultural heritage experts were asked to draw or to describe how they would imagine that the results should be presented. Secondly, we used a scenario-based technique, in which scenario’s of use were analysed to identify elementary or unit tasks, which were subsequently synthesised into basic user tasks (Rizzo et al., 1997). In fact, in this project the regular requirements-analysis approach failed because there were simply too many requirements, at least on the paper specifications. For designing the user interface, it did not at all matter that the system was intelligent or agent-based; that was merely the ‘service’, which delivered the result, just like the translation of terms was a particular service. Note that this perfectly fits the mashup architecture in new media designs. The same applies to e.g. the choice of the particular modality or the timing or manner of presenting information to the user in all of the projects concerned. As such, we assume that there is no principle difference between user interface design for new media or IoT applications and human-robot interfaces: the intelligence of the application, the user-adaptive and adaptable aspects, the modality and even the social-conventions of the interaction may all be designed and implemented as particular services in a client-server architecture, and indeed, even the user interfacing may be regarded as a specific service.

By way of conclusion, we would like to argue that interface design, regardless whether concern is with a textual interface, a talking head or a fully mobile and social robot, the design of the user or human interface is not principally different from regular user interface design. What is and remains a necessary requirement, particularly when we have to deal with any intelligent systems with their not-so-well-predictable outcomes is the requirement that, first, the design should allow for an flexible co-creation or co-design approach to ensure that the systems' behaviour fits the prospective users and, secondly, that it should allow for (agile) exploration and or 'tuning' of the design space to find the best possible way of presenting information or behaving, etc. to adapt the technical system to the user's context.

Agile design and design exploration facilitate that **the** design is less concerned with pre-specified requirements and more with creating working code (cf. the Agile Manifesto, see: www.agilemanifesto.org) or a usable system by learning from the actual usage of applications, for example in so-called "Living Labs" (cf. Chi, 2008).

CONCLUSION

This paper discussed the need for a user-centred, lightweight and flexible approach to design the user- or human interface of robotic and other intelligent systems, the form of a combination of user-centred design, co-design and co-creation. In our view it is often amazing how-well so-called 'ordinary users' are able to shape their own interaction with complex and intelligent systems, provided that they have been supplied with the proper tools and design facilitation.

REFERENCES

Chi, E.D. (2008). Living Laboratories: Rethinking Ecological Designs and Experimentation in Human-Computer Interaction. Augmented Social Cognition Research Blog from PARC. Available from: http://asc-parc.blogspot.nl/2008/11/living-laboratories-rethinking.html

Van Dijk, D., Kresin, F., Reitenbach, M., Rennen, E., and Wildevuur, S., (Eds.)(2011). Users as designers - a hands-on approach to creative research. Waag Society, Amsterdam.

de Haan, G. (2000). Interacting with a Personal Wearable Device. In: Wright, P., Dekker, S. and Warren, C.P. (eds.) Proceedings of ECCE-10. Human-computer Interaction: Confronting Reality, 2-15. August 21-23, Linköping, Sweden.

de Haan, G. (2003) The Design of I-Mass as a Tool for Interacting with Cultural Heritage. Tools for Digital Interaction, Int. Symposium on ICT 03, 24-26 Sept 2003, Dublin, Ireland.

de Haan, G., van der Mast, C.A.P.G., Blanson Henkemans. O.A. and Neerincx, M.A. (2005). SuperAssist: Personal Assistants for Diabetes Healthcare Treatment at Home. In: Sloan, A. (ed.), proc. HOIT 2005: Home-Oriented Informatics and Telematics. Springer, New York, 2005, pp. 261-275.

de Haan, G. (2013). A Vision of the Future of Media Technology Design Education - design and education from HCI to UbiComp. In: Proceedings Computer Science Education Research Conference - CSERC 2013, 4-5 april 2013, Arnhem, 66-72.

de Haan, G. (2015). HCI Design Methods: where next? From user-centred to creative design and beyond. Proc. of ECCE 2015: Understanding Design through Cognition. 1 – 3 July 2015; Warsaw, Poland.

Rizzo, A., Andreadis, A. and Marchigiani, E. (1997). The AVANTI project. In: Proceedings of ACM DIS Conference, Amsterdam, August 18-20.

Sanders, E.B.-N., and Stappers, P.J. (2008). Co-creation and the new Landscapes of Design. *CoDesign*, 4 (1), 5-18.

Schwaber, K., and Beedle, M. (2002). Agile Software Development with Scrum. Prentice Hall.

Weiser, M. (1991). The Computer for the 21st Century. Scientific American, 265(3), 94-104.