Adaptive Human-Machine-Interface of Automation Systems

Farzan Yazdi and Peter Göhner

Institute of Industrial Automation and Software Engineering, Pfaffenwaldring 47,70550 Stuttgart, Germany
{farzan.yazdi,peter.goehner}@ias.uni-stuttgart.de

Abstract. Automation systems are inseparable part of everyday life; heating systems, ticket vending machines or blood glucose meters are few examples of such systems, showing the diversity of their application domain. This diversity implies the variety of different user groups with assorted capabilities interacting with such systems in different contexts. Hence, the requirements of the human-machine interfaces of such systems are strongly varying, depending on the context of use. Attempts in developing high interactive systems, such as user centered development or universal design have failed; either they are costly or system specific. Furthermore, many context-relevant aspects are only known at run-time. In this paper, we propose a generic concept, which adapts the human-machine interfaces of automation systems at run-time, according to the context of use. It addresses not only the representational aspects but also the semantics and the connection to the underlying technical system. The concept is implemented as an evaluating prototype.

Keywords: human-machine-interaction, usability, adaptive user interface, context of use, context sensitive user interface, automation systems.

1 Introduction

The application growth of automation systems has brought a wider range of users among those who interact with such systems. Hence, future-oriented automation systems must satisfy a broad range of expectations. Furthermore, the demographic changes bring new requirements on automation systems to be used by users with limited capabilities. The new trends in the domain of automation systems are “Have-It-Your-Way” solutions for better usability [1] and cost reduction in maintenance, training, support, etc. [2]. **Usability** is “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use” [3]. This definition abstractly frames out the guidelines of ergonomics and usability of human-machine-interaction. Similar definitions used in the literature point out the same aspects as effectiveness, efficiency and satisfaction. While usability has been long discussed in the domain of web and smartphone applications, it is quiet new in automation systems domain. In some cases, the provided guidelines for the web applications can be adopted to automation system [4], however, automation systems have different requirements, imposed by their software-hardware architecture.

Automation systems are systems consisting of a technical process running in a technical system that is automated by components necessary for automation. These components can be sensors, actors and directly wired components to interact with the technical system. The automation functionality is realized on automation computers that are interconnected by a communication infrastructure. Finally, there are components to display information and to input user interventions to interact with the users of the system [5]. We refer to the last component as User Interface (UI). It includes not only the graphical UI but also the modalities, hardware and the interconnections with the technical system. The UI is exposed to the environmental conditions, HW/SW- circumstances and the users, referred to as Context of Use (CoU).

Addressing the usability issues, the researchers and engineers – both in academic and industrial areas – investigate methods, concepts and technologies to increase the usability of automation systems and to provide more intuitive and flexible interaction to the users. Principally, two categories of solutions can be determined:

- Development process-oriented for correct integration of usability requirements;
- Run-time-oriented to provide suitable interaction methods on demand.

Consideration of the development process has the advantage of providing the necessary system architecture, required for integrating the usability requirements. However, it can be very costly and cannot fully allow sensibility to all aspects that affect human-machine interaction at run-time. Furthermore, predefinition of different UIs suitable for different user groups and environmental conditions is impossible. Therefore, the run-time solutions is the reasonable alternative and the new trend in this field [6]. It has the advantage of being able to react to demands that are only known at run-time without the necessity to undergo costly analysis procedure of anticipating and forecasting different run-time situations.

2 Relationship to Collective Awareness Systems

This research project addresses the domain of automation systems and focuses on the enhancement of the usability of such systems. It refers to both industrial plant automation systems and product automation systems. The proposed concept adapts the UI of such systems according to the CoU. The ***development of such systems and the new trends and visions*** affect the proposed concept. The concept contributes to the development of ***computational and perceptional systems*** of collective awareness platforms. It provides the automation system with the ***intelligence to perceive the CoU*** and to react upon different circumstances. Furthermore, it proposes methods and technologies of ***industrial informatics*** for the necessary infrastructure.

3 Survey on Usability Issues and Existing Solutions in the Domain of Automation Systems

Automation system are in their operation very dependent on run-time parameters. Depending on the user's individual properties the quality of interaction can differ; e.g., user's experience or his/her disabilities. Additionally, environmental effects, such as background light or noise, can influence the interaction. These factors are examples of the CoU. CoU includes the user's profile, the tasks, means of work (hardware, Software and material) and the physical and social environment [7].

In order to increase the usability of automation systems, it is necessary to react upon different states of CoU [8]. One challenge is that the CoU can change dynamically at run-time; the computing platform might change, network bandwidth may alter and user role or capabilities can differ. Hence, a remedy at run-time is required. This has initiated a trend towards context-sensitive UIs. However, the categorization and capturing of the CoU, in a way that is readable for the machine is not trivial; the details of the affecting factors are domain and system specific. For instance, the affecting parameters in the domain of e-learning systems are different from those of automation systems. Many solutions regarding the capturing and representing the CoU have been proposed [9, 10]. However, there are no uniform and generic solution for reflecting the dynamic changes of CoU onto the UI [8]. Moreover, this reflection leads to unstable representation of UI. Therefore, the maintenance of usability is another important task to carry out [8]. Realization of context-sensitive UIs is another effort, which is often limited to the used technologies in the underlying automation system and the interaction tasks. Finally, the semantic clarity of the interaction tasks should be sustained in the adapted system.

The hardware-software architecture of the automation system is, besides CoU, relevant to the usability of the system, since it provides the interaction means. While at run-time the system architecture is hard to manipulate, often reconfigurations and deployment of system modalities can account for improvement of usability.

The existing run-time solutions can be divided into model-based, migratory or optimization approaches. Model-based approaches attempt to reduce the complexity of the CoU by abstracting it into models [11, 12, 13]. The models are structured with meta rules, defining the expected parameters. The models are then specified at run-time and are used by an automated component to generate the suitable UI. Some solutions predefine different configurations of UI and decide at run-time which configuration to use. This has the advantage that the implementation of such systems is simple and less computations are required. However, it requires thorough analysis of the CoU and is not flexible to new situations. Another method for generating the UI is by models that describe different aspects of UI [14]. The advantage here is the fact that depending on the amount of models used, the details of a UI can be defined to the granularity level needed. The problem here is that there is no unified approach to model UI. Specifically, in the domain of automation systems where real-time requirements are often the case, modeling of temporal behavior of the UI is non-trivial [15]. Moreover, generation of such models at run-time is a tremendous effort.

Migratory approaches deal with allocation and configuration of different system modalities and available interaction components [16]. With the popularity of smartphone, it is nowadays very common to perform tasks using multi-devices. Hence, an infrastructure to support cross-device and cross-platform interaction is necessary. Often web or cloud technologies are employed for exchanging data [17]. The migration can be done partially or fully, according to the requirements of the specific system [18]. Usage of multimodal concepts and embedding devices, which come along with the user (e.g. smartphones), are the advantages of these solutions. Users already know these devices; hence, the interaction is much simpler. However, the existing solutions are limited to software applications. Our tests on Programmable Logic Controller (PLC)-based and Controller Area Network (CAN)-based interaction systems, which are very common in the field of automation systems, showed that manipulation of the technical system from uncoupled devices are very hard to achieve. The platform dependencies and security issues make this even harder.

The optimization solutions are popular among computer science researchers. They mainly focus on the representation of the UI and according to CoU calculate all possible configurations/compositions of the UI. An optimization function traverses the possibilities and find the best solution to be exposed. The UI can be provided either using modelling techniques or using run-time interpreters, e.g. [19] or the smart toolbar grouping in Microsoft-Office 2007 or later. This category of approaches is scalable and very precise in its results. However, since they are very computation intensive, they are less suitable for the automation domain. Moreover, design of the optimization function is not trivial for so many parameters of the CoU.

4 Context of Use in Adaptive User Interfaces

The CoU in its initial definition from the ISO-Norm [3] is not specific enough. Regarding the goals of the concept and after thorough survey of the state of the art, we have summarized the following aspects of CoU, relevant for automation systems:

User Tasks: User tasks characterize the semantics of the interaction. They stand for the elements used on a UI and construct the sequence of activities needed to be done in order to complete a task [20]. There are four types of tasks [21]: User tasks, system tasks, Interaction tasks and Abstract tasks (A set of tasks, which can be grouped together in accordance to their semantic goals). This categorization is important, when reallocation of tasks by the user and by the automation system is necessary [22].

User Role: An automation system specifies a set of tasks (and consequently system functions) to a certain user role. Different user roles affect directly the type of tasks a user might probably intend to perform. For instance, maintenance staff, operator and system administrator are three user roles, which can indicate what a user belonging to a certain user role group would perform and what experience he/she has.

Users' Individual Properties (User Profile): [23] classifies user properties, regarding the context of use, in three categories: User role, user experience, knowledge and skills and personal attributes. [24] contemplates the human factor in its actions to perform everyday tasks. Here the user activities are divided into three abstract steps: receiving information, processing information and realizing the information. For receiving information, regarding the current technological possibilities, three human senses sight, hearing and touch are used. The cognitive capabilities of the user (fluid and crystallized intelligence) contribute to processing information and individual power, endurance and coordination capabilities correspond to realizing information. [25] investigates in a similar way as above the human factors in Human Machine Interaction (HMI). It sets categories of human factors and attempts to provide value ranges that define the normal values and their extents, using statistics and anthropometrics.

Environment: It is often in literature distinguished between physical environment and organizational environment. The environment (place) in which the interaction occurs is referred to as workspace. The physical environment characterizes the workspace conditions (i.e. atmospheric, auditory, thermal and visual conditions and the environmental instability), the workspace design (i.e. space and furniture, user posture and location) and the workspace safety (i.e. health hazards and required protective clothing and equipment) [7][23]. The organizational environment defines aspects such as the organizational structure or attitude and culture. These aspects can affect the HMI from a social and psychological point of view, hence, it will not be delved into any deeper here. [26] references five typical examples of environment influences, which can strongly affect the usability: Glare caused by bright light, environmental noise, physically limited workplaces, distractions and the tasks with special concentration demands.

5 The Concept of Adaptive User Interfaces

Based on the previous discussions and identified deficiencies of existing solutions, the concept has to fulfill the following requirements: 1) It must adapt the UI to the CoU. 2) The adaptation should occur at run-time and aims at existing systems (released systems). 3) It must consider the characteristics of automation systems. 4) It must be generic. The derived requirements stipulate abstractly the boundaries of the approach required to fulfill them. The concept has the goal to consider the existing automation system, i.e. automation systems on the market and to adapt and alter the user interface according to the CoU. Based on the definition of CoU given above, the adaptation should be *Task-oriented*, *User specific* and *Environment specific*. As mentioned earlier, the new trend to use mobile devices for the purpose of interaction is essential for the concept to be useful for current and future systems. We refer to the interaction devices – coupled or uncoupled to/from automation system – as Interaction component, hence, the adaptation should also be *interaction component specific*.

The task-oriented adaptation necessitates the computer-based analysis of the tasks at run-time. The challenges here are that the semantic description of tasks are difficult to capture at run-time [8] and that a description methods covering the requirements mentioned previously, including all necessary details and generic for the domain of automation systems does not exist. For the description method, we have proposed an XML-based description. Each task corresponds to an XML tag. The composite tasks are described as interlaced tags. Each tag have an ID and a task type as properties. For each task type – user tasks, system tasks, interaction tasks and abstract tasks – a keyword is dedicated. The temporal behavior of the tasks is adopted from [20], since it thoroughly describes all possible compositions of the tasks. The applicability of this proposed method has been evaluated using the prototype of a ticket vending machine. In addition to the tasks, the description of the original UI (UI before being adapted) is necessary for the adaptation; e.g., it is necessary to know which tasks belong to which UI-dialogue. UIML [27] is suitable for this purpose, since it not only describes the representation of the UI but also the connection to the underlying system functions. This is specifically interesting for event-based realizations of UI, which is often the case in automation system; e.g. refreshing a sensor value upon an interrupt. In addition, the discussed parameters of the CoU in previous chapter are the basis for the proposed concept.

System's performance can affect the usability when the interaction flow is disturbed. Therefore, the adaptation process has been divided into two stages: static and dynamic adaptation. This separation allows a pre-adaptation process to take place, before the high computational tasks start. The static adaptation refers to those parameters of CoU, which are static during an interaction session; e.g. user individual properties or user role. The dynamic adaptation takes care of parameters changing during an interaction session; e.g. light intensity or the interaction component. Figure 1 illustrates an overview on the concept, its components and the CoU.

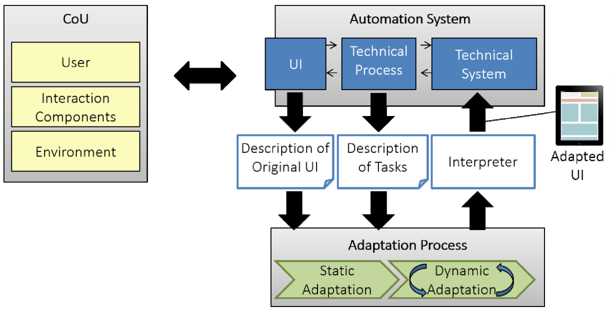

Fig. 1. Illustration of the concept its components and the CoU

A rule-based logic projects the situation constraints onto the UI arrangement. The rules are designed in three levels: 1) Modalities and their configuration; e.g. when user is visually impaired then visual interaction is not applicable. 2) Tasks and semantics; e.g. when user role x then load task set y or if user experience x then separate task y into sub-tasks. 3) Representation and layout; e.g. if display resolution less than x then group widgets. In the design of the rules, we have used the existing standards and guidelines on usability, e.g. [28]. The representation decisions rely on the method used to describe the original UI. It traverses the original UI and adapts each element of the description to the CoU. The result of the adaptation process is an adapted description of UI, which should be interpreted at run-time for different goal platforms. This makes the concept generic to different automation systems. To provide a multiplatform solution, we have proposed an interpreter, which from one side receives a unified description of UI, specified by the concept. From the other side it must be adapted to each goal platform, once in an initialization phase. In an HTML-based platform, a standard web-browser could act as an interpreter. The details of the interpreter is outside the scope of this paper. Hence, it will not be further explained.

6 Summary and Outlook

In this paper, the basics of usability of human machine interaction and CoU in the domain of automation systems were discussed. The deficiencies and shortcomings of the existing solutions has been used to motivate the investigations for a new solution. A view on relevant aspects of CoU provide the basis of the concept. The proposed concept of adaptive user interfaces is described. Various decisions on used methods in the concept has been made, based on experiments or implemented scenarios. The plausibility and conformity of the concept has been assessed using the prototype ticket-vending machine. It realizes multimodal/multiplatform interaction in two scenarios (operation and maintenance). The supported modalities are touch-LCD, speech input/output and gesture/mimic control. It provides evaluation of a broad range of interaction concepts. The platforms used in the system are Java, C#.NET and HTML5 with Jscript. In addition to the evaluation of the concept, regarding its plausibility and applicability, the prototype confirmed the necessity of a component to maintain the usability at run-time, upon changes of UI to the CoU.

The proposed concept assumes that the information on CoU are available. However, it is interesting, if the concept could work with incomplete information on CoU, which is often the case in reality. One idea to compensate the missing data would be to employ usage history or statistical information. This aspect is one of the milestones of our future works. Furthermore, it is interesting to compare the evaluation results with another system, with different characteristics, e.g. minimal performance systems or systems with complex internal communication system.

References

1. Nivethika, M., Vithiya, I., Anntharshika, S., Deegalla, S.: Personalized and adaptive user interface framework for mobile application. In: International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1913–1918 (2013)
2. Cooke, L., Mings, S.: Connecting usability education and research with industry needs and practices. IEEE Transactions on Professional Communication 48, 296–312 (2005)
3. ISO 29249 - Ergonomics of Human System Interaction (2006)
4. Rabin, J., McCathieNevile, C.: Mobile Web Best Practices 1.0 – Basic Guidelines. W3C Recommendation (July 29, 2008), `http://www.w3.org/TR/mobile-bp/`
5. Maga, C., Jazdi, N., Göhner, P.: Requirements on Engineering Tools for Increasing Reuse in Industrial Automation. In: 16th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2011), pp. 1–7 (2011)
6. Blumendorf, M., Lehmann, G., Albayrak, S.: Bridging models and systems at runtime to build adaptive user interfaces. In: Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems, pp. 9–18. ACM, New York (2010)
7. ISO 9241-11:1998, Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability
8. Vanderdonckt, J., Grolaux, D., Van Roy, P., Limbourg, Q., Macq, B.M., Michel, B.: A Design Space for Context-Sensitive User Interfaces. In: Proceedings of the ISCA 14th International Conference on Intelligent and Adaptive Systems and Software Engineering, Novotel Toronto Centre, Toronto, Canada, IASSE 2005, pp. 207–214 (2005)
9. Calvary, G., Coutaz, J., Thévenin, D.: Embedding Plasticity in the Development Process of Interactive Systems. In: Proceedings of Workshop on User Interfaces for All (2000)

10. Crease, M., Brewster, S., Gray, P.: Caring, Shar-ing Widgets: A Toolkit of Sensitive Widgets. In: Proceedings of BCS Conference on Human Computer HCI 2000, pp. 257–270. Springer, Berlin (2000)
11. Vellis, G., Kotsalis, D., Akoumianakis, D., Vanderdonckt, J.: Model-Based Engineering of Multi-platform, Synchronous and Collaborative UIs - Extending UsiXML for Polymorphic User Interface Specification. In: 16th Panhellenic Conf. on Informatics, pp. 339–344 (2012)
12. da Silva, P.P.: Object modelling of interactive systems: the UMLi approach. Thesis sumbited to the University of Manchester for the fegree of Doctor of Philosophy in the faculty of science and engineering (2002)
13. Trætteberg, H.: A hybrid tool for user interface modelling and prototyping. In: Proceedings of the Sixth International Conference on Computer-Aided Design of User Interfaces CADUI 2006, Bucharest, Romania, June 6-8, ch. 18. Springer, Berlin (2007)
14. Meixner, G., Seissler, M., Breiner, K.: Model-Driven Useware Engineering. In: Hussmann, H., Meixner, G., Zuehlke, D. (eds.) Model-Driven Development of Advanced User Interfaces. SCI, vol. 340, pp. 1–26. Springer, Heidelberg (2011)
15. Nóbrega, L., Jardim Nunes, N., Coelho, H.: Mapping concurTaskTrees into UML 2.0. In: Gilroy, S.W., Harrison, M.D. (eds.) DSV-IS 2005. LNCS, vol. 3941, pp. 237–248. Springer, Heidelberg (2006)
16. Bandelloni, R., Paterno, F., Salvador, Z.: Dynamic discovery and monitoring in migratory interactive services. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, pp. 603–607 (2006)
17. Raising the Floor Community, Global Public Inclusive Infrastructure (GPII), http://gpii.net/
18. Ghiani, G., Paternò, F., Santoro, C.: User Interface Migration Based on the Use of Logical Descriptions. In: Migratory Interactive Applications for Ubiquitous Environments. Human-Computer Interaction Series, vol. 2011, pp. 45–59 (2011)
19. Gajos, K.Z.: Automatically Generating Personalized User Interfaces. PhD thesis, University of Washington, Seattle, WA, USA (2008)
20. Paternò, F., Mancini, C., Meniconi, S.: ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models. In: IEEE Proceeding INTERACT 1997 Proceedings of the IFIP TC13 Interantional Conference on Human-Computer Interaction, pp. 362–369 (1997)
21. Paternò, F., Meixner, G., Vanderdonckt, J.: Past, present, and future of model-based user interface development. i-com Journal for Interactive und Cooperative Media 10(3), 2–11 (2011)
22. Schweitzer, G.: Mechatronics for the Design of Human-Oriented Machines. Transactions on IEEE/ASME Mechatronics 1, 120–126 (1996)
23. Maguire, M.: Context of Use within usability activities. J. Human-Computer Studies 55, 453–483 (2001)
24. Hentschel, C., Wagner, A., Spanner-Ulmer, B.: Analysis of the application of the assembly-specific evaluation method EAWS for the ergonomic evaluation of logistic processes. In: Annual International Conference of the Institute of Ergonomics and Human Factors, pp. 221–226. CRC Press, Taylor & Francis, London (2012)
25. Federal Aviation Administration, Human Factors Division: FAA Human Factors Awareness Web Course, https://www.hf.faa.gov/Webtraining/index.htm
26. ISO 9241-20:2008, Ergonomics of human-system interaction – Part 20: Accessibility guidelines for information/communication technology (ICT) equipment and services
27. Abrams, M., Phanouriou, C., Batongbacal, A.L., Williams, S., Shuster, J.: UIML: An Appliance-Independent XML User Interface Language. In: Proceedings of 8th International World-Wide Web Conference WWW'8. Elsevier Science Publishers (1999)
28. World Wide Web Consortium W3C, Web Content Accessibility Guidelines (WCAG) 2.0 (2009), http://www.w3.org/Translations/WCAG20-de/