Real-Life Guards for Human-Centered Systems

Nikola Serbedzija
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
D-10589 Berlin
nikola.serbedzija@fokus.fraunhofer.de

ABSTRACT
With increasingly smart consumer electronics, the human role in computing shifted from an information user to a subject of computation. Accordingly, protective mechanisms are required to look after "the man in the loop". The paper suggests a strategy to prevent negative technology impacts by including real-life guards in early design phase.

CCS Concepts
• Human computer interaction (HCI) • IoT.

Keywords
Neuro processing, physiological computing, real-life computing.

1. INTRODUCTION
Fast development of cyber-physical and IoT [1] systems and their wide acceptance in praxis are changing our everyday life, affecting our individual and group behaviors, our privacy and socializing. Increasingly, authors are expressing their concern warning that " we need to be thinking deeply now about broad IoT use and deployment, and how it can help create a more enlightened and civilized society. If we wait too long, we do so at our own risk"[2].

This paper focusses on design strategies to ensure side effect free and real-life conform system functioning. It assumes that most of the unwanted "side-effects" could be prevented, if considered in early design phase.

2. HUMAN-CENTRED SYSTEMS
Human centered computing is a very sensitive domain where technology impact may be particularly harmful [3]. In case of purely technical systems, the scientists have argued that the potential for damaging effects lies in application, not in technology itself. The argument is hard to deny as indeed, elegant and generic scientific solutions inherently allow for a wide range of uses. It is the actual application that makes the technical concept good or bed, moral or immoral, positive or negative, not the concept itself.

However, this plausible excuse, which relieves scientists' conscience, cannot be directly applied in the rapidly emerging human-centered systems that in a cross-discipline manner correlate humans with computing artifacts. Here, the burden of responsibility is not clear-cut. It cannot be taken from the scientists, leaving them in a comfortable, ethically neutral position, and entirely shifted to notably less conscientious, business-driven application providers. The potential perils must be perceived, reflected upon and eventually prevented, not only in deployment phase, but also in the course of system design and development. By making moral judgements and by providing means to encode them in the conceptual design, the scientists will take their share of responsibility and make the technology influence balanced and sensitive to personal, social and ethical concerns.

PETRA '17, June 21-23, 2017, Island of Rhodes, Greece
ACM 978-1-4503-5227-7/17/06.
http://dx.doi.org/10.1145/3056540.3076210

The human-centered systems take into account biological, personal, behavioral and social aspects of human being in order to provide seamless man-machine confluence. Alternatively they are referred to as human-in-the-loop computation, as they place humans in the center of the processing, relying not on explicit but rather implicit, even unconscious man-machine interaction.

2.1 Human-in-the-Loop
Human-in-the-loop systems are equipped with a number of sensors and actuators used to collect information about individuals and the environment and affect real-life situation or activity. Application domains range from brain-computer interface [4], that uses different sensors to measure brain signals and decode certain mental states and use them as "implicit" control commands, via physiological computing [5], that extracts significant psycho-physiological markers used to diagnose psychological states and control some environment or tasks accordingly, up to social media apps that enhance our communication abilities based on a numerous information about the users location, preferences, habits, contacts, etc. collected by smart phone sensors. Their common characteristic is seamless and sensor-rich data acquisition, personal and situational analyses and personalized control. Applications are mind typewriter, neuro prostheses, affective player, mentally driven video games, neuro-feedback systems, social networks, etc.

The common feature of these systems is a loop control where system (1) collects data with specific sensors; (2) analyses the personal state and surrounding situation and proposes some action; (3) influences the person and environment via adequate actuators (tactile, audio-visual, etc). The loop processing cycles, with each iteration fine-tuning the previous one. At one side, the applications do good to their users without any explicit users' instruction or their conscious participation, at another, the devices collect personal information and may distribute them further unnoticed and without users consent. Consequently, the encoded manipulative strategy can be beneficial but also harmful for the human in the loop.

2.2 Technology Impact
In all closed loop control systems, which are traditionally used to govern some technical process, the controlled process has to be thoroughly understood. All aspects that define the controlled phenomenon (a physical process) and its dynamics, as well as the timely system response have to be taken into account from the specification and design up to development, testing and deployment phase. If the system failed to respond to non-functional

requirements – it would not be used in practice, as it could harm the very object of computing. Surprisingly, in human-centered systems this thorough consideration of the controlled "object" has been neglected.

What differs human-centered systems from standard control systems is the very object of computation: it is not a physical process, but a subjective and adaptable species. That very ability for adjustment creates a possible shortcut for all kind of influences. When exposed to new technology we do change, since we modify our norms and behaviors (e.g. sacrificing privacy for obtaining certain functionality). This is in the long run detrimental, as it makes us more dependent and more vulnerable. Especially when the human-centric systems ignore the complexity of our nature, our biological, psychological and social rhythm and our capability to adapt, non-wanted long term impacts can be particularly harmful.

We strongly believe that human-centered computing requires novel technical approaches that safe guard the subject of computation, taking into account not only system functional features, but also non-functional ones, primarily compliance with human psychophysical condition, social and ethical norms. In that respect, pervasive technology designers have to include the ethical and behavioral principles into consideration in early design phase and encode them in their system.

3. REAL-LIFE COMPUTING

Real-life computing is a concept that aims at including social responsibility in software engineering by providing a systematic protection of potentially damaging impacts. Safeguarding strategies for human-in-the-loop systems are researched in order to guarantee ethically conform and side-effects-free system functioning in a provable manner. The approach refines the development lifecycle by the inclusion of non-functional constraints (like safety, privacy and mental state protection, etc.) in all lifecycle phases – from requirements specification up to deployment and testing. It is done by providing software abstractions that support adaptive close-loop control – the driving processing model of human-centered systems.

The bio-cybernetic loop [6] is a concept that supports closed-loop control for human-centered systems. It is a cyclic processing mode that includes: (1) personal data acquisition; (2) specific state/situation evaluation, (3) implicit system generated stimuli. These three major phases iterate in an endless loop driven by high-level strategies that should re-inforce specific mental or physical function or state. The bio-cybernetic loop operates at several levels of processing, forming a hierarchy of internal loops.

The existing component-based middleware that runs multiple bio-cybernetic loops featuring pervasive adaptation [6] is enriched by knowledge components that allow for adaptive and self-corrective processing. To behave autonomously, a control component maintains knowledge about itself (its objectives, capabilities, execution state and restrictions) and about its execution context – real-life constraints (situations/states that may endanger the subject). Such collection of facts yields system's awareness of the own functionalities and effects it has on the subject, which allows for adaptive protective run-time behavior.

To deploy real-life concepts in a systematic way, a programming abstraction called real-life guard (RL-guard) has been designed and developed. RL-guards are included at each hierarchical level of the bio-cybernetic loop components, specifying different levels of protection.

The idea behind RL-guards is to achieve a transparent mechanism that is completely separate from the application logic, but is able to intercept the execution at any time and inforce the protective measures, providing the conformance with real-life concerns. It is effective way to deal with "negative impacts" as it decouples the detection and handling from regular execution flow. At one side, it makes the control program cleaner (as it focuses on the pure functionality) and at another it provides means for automatic discovery and propagation of faulty conditions from the point of occurrence to the point of recovery (handling).

A typical real-life computing development process that includes real-life protection has the following flow: (1) formal specification of "the conditions to be prevented" (design phase), (2) creation of software abstractions for RL-guards (development phase), (3) run-time support for continuous monitoring of RL-guards (in deployment phase), (4) triggering of automatic recovery (dynamic strategy change) in case a negative impact occurs (in run time and in real life). The RL-guard mechanism can be imagined as a well-known exception handling mechanism. Here, however, the exception is not the event that may crush the execution flow and affect the processing logic, but an unpredictable notion of "damaging impact on the user" which may harm the subject of processing, namely the human involved.

With these built-in principles, bio-cybernetic loop relies on real-life experience and is able to self-improve and respond with strategy changes, when the subject of computation is exposed to a potentially harmful situation. The idea behind the protective mechanism is the system ability to discover possible violation of real-life constraint before it really occurs. That enables the system to perform automatic recovery action, rather than to leave the person to deal with possible consequences of the system miss behavior.

4. CONCLUSION

The paper presents a rationale for a new computing paradigm, called real-life computing that equips the human-centered systems with protective-by-design mechanisms to guard the persons involved in computation. The concept is general-purpose and applicable in most of systems that place humans in the center of processing loop. An experimental framework that features real-life computing is developed as an extension of existing reflective middleware [6], enriched to support RL-guards.

5. REFERENCES

[1] Li, S., Xu, L.D. & Zhao, S. (2015). The internet of things: a survey. Information Systems Frontiers, April 2015, Volume 17, Issue 2, pp 243–259.

[2] Berman, F. and Cerf, V.G. 2017. Social and Ethical Behavior in the Internet of Things, Communication of the ACM, Vol.60, No. 2, Feb. 2017, pp.6-7.

[3] Serbedzija, N. 2015. Autonomous Systems and Their Impact on Us. *Lecture Notes in Computer Science,* vol. 8950, Springer Verlag, Heidelberg, 662-675.

[4] Peck, E., Carlin, E. and Jacob, R. 2015. Designing Brain−Computer Interfaces for Attention-Aware Systems. *IEEE Computer*, October 2015, pp34-42.

[5] Fairclough, S. 2015. Physiological Computing, *IEEE Computer*, October 2015, pp 13-15.

[6] Serbedzija, N and Fairclough, S. 2012. Reflective Pervasive Systems. *ACM Transactions on Autonomous and Adaptive Systems (TAAS*), Vol. 7 (1), 12:1-19