Social Sensing: When Users Become Monitors

Raian Ali, Carlos Solis,
Mazeiar Salehie, Inah Omoronyia
Lero - University of Limerick, Ireland

Bashar Nuseibeh
Lero- University of Limerick, Ireland
The Open University, UK

Walid Maalej
Technische Universtität
München, Germany

ABSTRACT

Adaptation requires a system to monitor its operational context to ensure that when changes occur, a suitable adaptation action is planned and taken at runtime. The ultimate goal of adaptation is that users get their dynamic requirements met efficiently and correctly. Context changes and users' judgment of the role of the system in meeting their requirements are drivers for adaptation. In many cases, these drivers are hard to identify by designers at design time and hard to monitor by the use of exclusively technological means by the system at runtime. In this paper, we propose *Social Sensing* as the activity performed by users who act as monitors and provide information needed for adaptation at runtime. Such information helps the system cope with technology limitations and designers' uncertainty. We discuss the motivation and foundations of Social Sensing and outline a set of research challenges to address in future work.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques
General Terms
Design, Human Factors, Management
Keywords
Requirements Engineering; Social Software Engineering; Models at Runtime; Adaptive Software Engineering.

1. INTRODUCTION

Self-adaptive systems are increasingly expected to cope with the volatile nature of the environment in which the system operates. Different categories of environmental changes trigger different categories of responses [1]. For example, security breaches and attacks could trigger certain self-protection actions, or changes in the available resources could trigger self-optimization actions. The ultimate goal of this so-called *self-** computing paradigm is that users' dynamic requirements are met efficiently and effectively, and adaptation is done autonomously by the system so that computing transparency is maximized and humans' (designers and users) effort is minimized [2].

The adaptation loop [3] consists of monitoring changes in the system operational environment, analysis of changes, planning an action, executing it, monitoring back the effects, and so on. Focusing on the monitoring stage, the system should monitor its context, i.e. the state of the environment in which it operates [4]. Moreover, the system has to monitor if its executed actions were performed successfully. Self-healing deals with incorrect execution in a way that allows a system to handle faults and errors autonomously. However, the technical correctness of system execution (bug-free, no connection errors happens, etc.) does not necessarily mean that users' requirements are met [5]. For example, sending an invitation to a meeting can be done via one of two system alternatives: by SMS or email. A successful sending of an invitation to a meeting via email does not necessarily mean that the invitee was notified on time as the invitee might miss the email or misinterpret it. That is, monitoring should primarily be concerned with determining if users find the system execution a valid and effective way for reaching their requirements, and adaptation should respond to how users judge each system execution against the meeting of their requirements.

Monitoring context changes and the quality of each system alternative is not always achievable with the use of solely technological means and might require users to collaborate with the system. For example, in a driver-assistant system, the traffic level in the area is a context attribute that affects to which park the system should guide the driver. Such context might be un-monitorable due to the lack of necessary infrastructure. As a solution, the system could rely on the information obtainable through the drivers' community in that area. The system could have different alternatives to interact with a driver while assisting him (voice commands, maps, street view, etc.). For instance, a quality attribute such as "readability" could be judged differently in different contexts for each of these alternatives. However, neither the designers at design time nor the system at runtime can decide with certainty how the drivers judge "readability" for each alternative. As a solution, drivers may be asked to provide such quality judgments at runtime after an alternative is executed.

Besides monitoring the values of context attributes and quality attributes, users could also be involved in identifying such attributes. Users act as monitors to decide relevant context and quality attributes to add to the design of the system and irrelevant ones to remove from it as well. For example, drivers might add "straightness of the road" as a context attribute which influences the quality of each interaction alternative (voice command, map, street view, etc.) against the quality attribute "readability". Moreover, drivers might add "minimum noise" as a relevant quality attribute, which the designers did not consider when designing the system, so that each system alternative is also qualified against it. Thus, users are also monitors for identifying drivers for adaptation, i.e. mainly context and quality attributes.

Maalej et al. [6] discuss how to make the user's involvement a first order concern in software projects, moving from a transactional to a social engineering process. In line with this view, the involvement of users can also be done at runtime as an integral part of the system operation and not only the engineering process. Ali et al [7] propose to weave together the variability of context and the space of alternatives designed to reach the requirements. However, context is presumed monitorable by the system at runtime and the relation between context and alternatives is specified under certainty. These two design assumptions are hard to achieve in certain systems, which might need humans to monitor context and its influence on the activation, adoptability, and quality of each system alternative.

ESEC/FSE'11, September 5-9, 2010, Szeged, Hungary.

In this paper, we propose *Social Sensing* as a system development technique which involves users, at runtime, in the monitoring activity of the adaptation loop. The goal is that limitations of technological devices as well as uncertainty and incompleteness of the system design are faced via the involvement of users' perception as an integral part of the system monitor. Social Sensing treats users as a primitive component of the system instead of pure consumers of its functionalities. We discuss Social Sensing foundations in Section 2, list research challenges in Section 3, and conclude the paper in Section 4.

2. SOCIAL SENSING: FOUNDATIONS

Social Sensing is based on exploiting users' perception as an integral part of the computation. The system relies on the users' community to get information which is un-monitorable by automated means and/or unspecifiable under certainty by designers at design time. The users play the monitor role and provide input to the system so that the right decision and response will be planned and enacted during the operation. This is particularly important when dealing with systems involving a community of users. For example, when volunteer drivers provide context information, e.g. the traffic level in a specific area, other drivers will benefit from it when the system executes for them. The information provided by the volunteer drivers about the quality of a system behavior, e.g., the comfort level of each interaction technique for guiding a driver, is the main ingredient for the collective judgment of the drivers' community about each alternative so the system can act accordingly.

We discuss Social Sensing in the context of adaptive systems. In such systems, the context monitoring as well as the validity and quality of system alternatives are essential to guide adaptation. Moreover, we focus on the problem space rather than the solution space taking the users' satisfaction about the role of system in meeting their requirements as the main goal of adaptation.

Social Sensing advocates that users can play a role in establishing the monitoring process. Users understand the system as a means to solve their problems and can collaborate with it as monitors providing information using their own terms, which belong to the problem domain (requirements, quality, context, validity, etc.) not the technical solution domain (bug, error, protocol, proxy, etc.). Thus, one of the ideal domains of Social Sensing is requirements-driven adaptation. In the rest of this section, we discuss the meta-model of this domain (represented in Figure 1) as a baseline for our Social Sensing method.

Variability is the cornerstone for adaptation. A system provided with only one alternative is unable to adapt when context changes. A system alternative is a synthesis between automated and human activities intended to reach certain requirements. In adaptive systems, a requirement could be reached via different system alternatives. For example, considering the driver-assistant system, the system could have two main system alternatives "guide to a public park" and "guide to a paid park". The interaction with a driver for guiding him to a suitable park can be also achieved via different alternatives such as voice commands, an interactive map, or a street view. Adaptation is seen as the selection of the system alternative which best fits to the current context. The fitness of a system alternative is measured via both its validity as a means to reach the requirements and its quality degree as well.

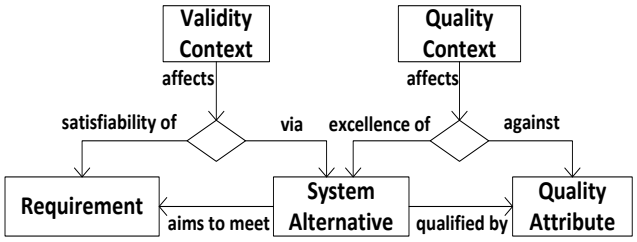

Figure 1 Meta-model of Requirements-driven Adaptation Artifacts

The validity of a system alternative is a binary property referring to its success/failure in reaching the requirement it is intended for. For example, using the guidance of the driver-assistance via one alternative, the driver either reaches (valid) or does not reach (invalid) a free parking place. The quality of a system alternative is captured via a number of quality attributes each representing a distinguished characteristic of the degree of excellence of an alternative. For example, the quality of each way of interacting with a driver could be refined to "readability", "fast", or "less distraction". The assessment of a system alternative against a quality attribute could fall into a designated scale (e.g. [very poor, poor, acceptable, good, very good], or [low, medium, high]). The validity and the quality of the operation of a system alternative in the past are main factors to consider when adaptation is planned so that the best alternative will be selected and applied.

The validity and the quality of a system alternative are context-dependent. Context is represented via context attributes, each one representing a distinguished characteristics of the environment in which the system operates, e.g., driving speed, driver age, traffic level, etc. Certain context attributes might influence the validity of a system alternative and/or its quality against certain quality attributes. For example, suppose the following context attributes (Driver is in a *hurry*, The distance to the public park is *far*, Traffic level is *high*) then most probably the system alternative "guide to public park" is invalid. Certain context attributes might influence the quality assessment of a system alternative against certain quality attribute. For example, the level of driving experience, the complexity of the road and the traffic level in the area are context attributes which influence the assessment of each alternative of communicating with a driver against a quality attribute like "less distraction".

Social Sensing plays a major role within the above settings and is characterized by the following four distinct contributions:

1. **Context values**. Users play a role in obtaining values of context attributes that affect the validity and quality of system alternatives, which are not monitorable for reasons such as limitations or failure of technology, lack of infrastructure, etc. Using these values, a system can decide applicable alternatives by analyzing the history of each alternative in similar values of context in the past. As a result, the alternative which best fits the current values of context will be applied. For example, a context attribute like "there is an accident in a certain area" may not be monitorable by the driver-assistant system due to the lack of access to the official traffic management system or because no such system exists. Thus when volunteer drivers passing close to the accident's location provide such information, the system will benefit from it for guiding other drivers.

2. **Quality and validity assessment**. Uncertainty is inherent when designing a system. The validity of a system alternative and its quality assessment against each quality attribute is not always decidable under certainty by designers at design time. In Social Sensing, the users play the role of monitors of the validity and quality of each system alternative. For example, whether guiding a driver with medium driving experience via an interactive map is a valid interaction method, is unknown unless the system operates in practice and drivers themselves decide that. Moreover, designers might not be able to decide the quality of guiding a driver, who is familiar with the area, via voice commands against the quality attribute "less distraction". Moreover, validity and quality are not static properties. What is known to be a valid and high-quality alternative at one point in time may lose these characteristics as time passes. Social Sensing allows for a continuous evaluation of the system alternatives by involving the users' community. For example, "voice recognition" might be judged as low quality interaction alternative compared to a quality attribute such as "ease of use" by drivers. In the future, when the drivers become more familiar with this technology, their judgment of its quality might be different. Social Sensing allows for capturing changes in the users' community judgment of the system alternatives so that adaptation is up-to-date.
3. **Context attributes identification**. Uncertainty concerns also the identification of the context attributes which affect the validity and the quality of each system alternative. Designers might be uncertain if their identification is correct and complete. Social Sensing allows users to act as designers while the system is operating, by dropping context attributes that they judge to be irrelevant and adding others which they believe to be relevant for the validity and the quality of each system alternative. In Social Sensing, users can engage with this process throughout the life of a system. This is essential to cope with the fact that relevance itself is not a static property and what is judged to be relevant at the moment might become irrelevant in the future, and vice versa. For example, unlike the designers' specification, the drivers' community might identify "the existence of a staff assistant" as a relevant context attribute that affects the quality attribute "reliability" of the system alternative "guide to paid parking". However, this attribute may turn out to be irrelevant when the drivers' community becomes more competent about the use of new technology and trusts it more. Moreover, the designers might specify that "the existence of traffic lights inside the park" is a context attribute which affects all alternatives against the quality attribute "less distraction". On the other hand, this decision might be seen as a wrong one by the drivers' community and they may decide collectively to drop this context attribute and consider it irrelevant.
4. **Quality attributes identification**. Similarly to the above discussion about context attributes, designers might miss quality attributes which the users' community finds relevant. Also, designers might include quality attributes that may be deemed irrelevant by the user community. Social Sensing gives users a voice and allows them to be a part of the decision making team. It allows them to continuously play the role of monitor to decide relevant quality attributes to add and irrelevant ones to drop when appropriate. For example, "reduced pollution" might be considered by the drivers' community as a relevant quality attribute when evaluating each system alternative so that the system might choose park place that is not ideal in terms of time and effort required to reach it but good for reducing the pollution in the area. Thus, if the drivers' community decides that this attribute is relevant, it will be added to the list of quality attributes defined initially by the designers. Moreover, the users' community might drop some attributes from that list if they are found to be irrelevant. For example, in a city where traffic is often low and the need to reduce pollution is not critical, the drivers' community might collectively decide to drop the attribute "reduced pollution".

Social Sensing allows users to express their opinion so the system analyzes it and takes decisions which reflect the collective intelligence of the users' community. The information provided by the users' community at runtime is a main ingredient for planning and enacting adaptation. On the one hand, it helps the system to cope with the limitations of the technological means of monitoring the environment and the uncertainty and incompleteness in the designers' decisions. On the other hand, it allows the users to drive the adaptation and maximize its correctness so that their requirements are reached in the best available way when changes happen.

3. RESEARCH CHALLENGES

While Social Sensing is powerful for crowd-sourcing users and enabling them to act as monitors, it brings several software engineering challenges.

1. **Users' subjectivity**. Social Sensing relies on the existence of a certain degree of similarity in the perception of different users. That is, Social Sensing requires that the perception of users of the values and relevance of the adaptation drivers (context and quality attributes, etc.) are similar. However, this is not always the case and users might perceive adaptation drivers subjectively. For example, the value of a context like "traffic level" could be monitored by one driver as "medium" and by another as "high". The same subjectivity could arise when assessing the system alternatives against a quality attribute. Devising methods and analysis mechanisms to normalize the different users' perception is a challenging problem of Social Sensing.
2. **Trust management**. Social Sensing requires users to provide information and thus implies dealing with trustworthiness of information and users. The benefits of the openness-to-the-crowd might be sacrificed if untrusted users, who might intentionally or unintentionally cause harm to the system or misuse it, are not detected and dealt with. Moreover, users need to trust the system itself before collaborating with it. Developing systems that adopt Social Sensing and are able to inspire users' trust is another socio-technical challenge, and achieving such trust has to be engineered as a first class requirement of the whole system.
3. **Security and Privacy**. Depending on the criticality and sensitivity of monitored information, security goals such as confidentiality, integrity, and availability might become concerns in Social Sensing. For example, Social Sensing might not be ideal for a driver-assistant system for an ambulance that is typically assigned to critical missions, unless information provided by the driver's community in the area is strictly verified and secured. Moreover, while

Social Sensing relies on crowd-sourcing a large number of users, it also opens the door for malicious users to attack the system. For example, some drivers might provide wrong information that leads to less traffic in the areas where they drive. Furthermore, it is notable that some of the security requirements in Social Sensing could be in conflict with other categories of requirements such as privacy ones. For example, if a driver refuses to provide his location for privacy reasons, the system might not be able to help him avoid traffic, thereby making the main system service practically unavailable.

4. **Transparency**. An important goal of adaptation is to minimize humans' (users and designers) effort and maximize computers' transparency. Social Sensing implies the intervention of users as monitors and thus users are required to provide input not necessarily used for their own immediate benefit. This means that Social Sensing, if not designed effectively, may provide adaptation capabilities for one group of people while potentially violating adaptation of another. For example, the evaluation of a system alternative may be provided by a user after the operation terminates, so that the system benefits in next operations executed for benefit of different drivers. Devising mechanisms to encourage users to act as monitors and feel some gain by doing this task is therefore a research challenge to address.

5. **Volatility.** The validity of information provided by users, especially context changes, is volatile. Context may change rapidly so that information, which was true when the users provided it, might become false when the system starts to plan and enact adaptation. Social Sensing design has either to deal with this volatility or to avoid taking decisions based on information having highly volatile validity. For example, when the car needs to be refueled, the driver-assistant activates the requirement "guide the driver to filling station". When the system receives information from other drivers that there is a filling station close to the driver location and starts to plan and execute adaptation (notifying the driver, getting his confirmation, choosing the right interaction method, etc.), the driver would have passed the station. That is, the system has to deal with the liveness of sensed information.

6. **Implementation**. There are major challenges regarding the implementation of Social Sensing. These challenges include the way to represent context and quality attributes and the values and judgments provided by users, the way to capture this information efficiently and independently from the applications, the decision about what information to collect exactly and how long this data should be stored, etc [6]. Moreover, involving users in dealing with large volume of information might compromise the applicability of Social Sensing. For example, the list of quality attributes provided by the users' community could increase to an extent where users find it tedious to assess a system alternative against all attributes included in it. This means that the system might need iterative maintenance so that applicability is not sacrificed. Ideally, the system has to help designers when maintaining the system by pointing out loci where designers need to fix errors or take some other altering actions.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed Social Sensing as a system development technique in which users' perception is part of the system computation. We advocated that users are a powerful source for information that drives adaptation. In Social Sensing, users act as monitors, increasing the ability of the system and designers for capturing the values and the relevance of certain adaptation drivers. Out of these drivers, we discussed context, quality and validity. Social Sensing implies a direct interaction with users. Thus, users interact using their own terms, i.e. their problem domain terms, and this explains the focus of our discussion of Social Sensing from a requirements engineering perspective.

Our future work includes developing a methodology (models, development process, analysis techniques, and a software framework for Social Sensing) for incorporating the role of users in the design of requirements at runtime. Our ultimate goal is to develop capabilities that make Social Sensing viable and useful from two perspectives. First, the users' interaction with the system should be facilitated and the awareness of users about the consequences and the benefits of their interaction should be maximized. In other words, engineering the awareness of users and facilitating and encouraging their collaboration with the system represent the first main thread of research we plan to conduct. Second, the system has to be provided with analysis techniques to process the information gathered from its users' community and make use of it at runtime. These include deciding about the significance of information provided by users and formulating the community's collective judgment, autonomously or with a minimum intervention of designers.

ACKNOWLEDGMENTS. This work has been partially funded by the EU Commission through the FastFix project, and by Science Foundation Ireland grant 10/CE/I1855. We also thank Vinny Cahill, Siobhán Clarke and Gavin Doherty for the discussions which enriched the idea presented in this paper.

5. REFERENCES

[1] Mazeiar Salehie, Ladan Tahvildari. Self-adaptive software: landscape and research challenges. ACM Transactions on Autonomic and Autonomic Systems, 4:2, pp. 1-42, 2009.

[2] Richard Murch. Autonomic computing. IBM Press, 2004.

[3] Robert Laddaga. Self-adaptive software. Tech. Rep. 98-12, DARPA BAA. 1997.

[4] Anthony Finkelstein, Andrea Savigni. A framework for requirements engineering for context-aware services. In STRAW 2001.

[5] Raian Ali, Fabiano Dalpiaz, Paolo Giorgini, Vitor E. Silva Souza. Requirements Evolution: From Assumptions to Reality. In EMMSAD 2011.

[6] Walid Maalej, Hans-Jörg Happel, Asarnusch Rashid. When Users Become Collaborators: Towards Continuous and Context-Aware User Input. In OOPSLA 2009

[7] Raian Ali, Fabiano Dalpiaz, Paolo Giorgini. A goal-based framework for contextual requirements modeling and analysis. Requirements engineering, Springer, 2010, 15, 439-458.