Six Modes of Proactive Resource Management: A User-Centric Typology for Proactive Behaviors

Antti Salovaara, Antti Oulasvirta
Helsinki Institute for Information Technology
P.O. Box 9800, 02015 HUT, Finland
{antti.salovaara, antti.oulasvirta}@hiit.fi

ABSTRACT
Proactivity has recently arisen as one of the focus areas within HCI. Proactive systems adhere to two premises: 1) working on behalf of, or *pro*, the user, and 2) acting on their own initiative. To extend researchers' views on how proactive systems can support the user, we clarify the concept of proactivity and suggest a typology that distinguishes between 6 modes of proactive resource management: preparation, optimization, advising, manipulation, inhibition, and finalization of user's resources. A scenario of mobile imaging is presented to illustrate how the typology can support the innovation of new use purposes. We argue that conceptual developments like the one proposed here are crucial for the advancement of the emerging field.

Author Keywords
Proactivity, proactive computing, ubiquitous computing, context, user-centered design.

ACM Classification Keywords
H5.2. User Interfaces: Theory and methods.

INTRODUCTION

One of the promises of context-aware systems is their ability to act without explicit instruction from the user. This ability can be used to provide beneficial information and make decisions on behalf of the user. One particular form of context-awareness, *proactivity*, has recently drawn more attention in context-aware computing (e.g., [2], [12–13]). Proactive systems adhere to two premises: 1) working on behalf of, or *pro*, the user, and 2) acting on their own initiative. However, not every automatically triggered action should be called proactive, or otherwise the term's usefulness is in danger. Moreover, researchers have not yet tackled the hard problem of what proactivity is *for* [9].

We believe that elaborating concepts, theories, models, and other thinking tools is necessary especially in the early stages of research. Therefore, in this paper, we briefly investigate the historical background of the concept, characterize and clarify it, and present a systematization of proactive behaviors.

Historical Context

The idea of a computer working as an autonomous partner with humans emerged already in 1960 in the vision of *man–computer symbiosis* by J.C.R. Licklider [6]. More recently, in the early 1990's, *UI agent research* devoted effort to achieve this symbiosis and understand its role in helping the user. Proposed ways included providing information to facilitate decision-making [10], hiding complexity of tasks, performing tasks on user's behalf, training and teaching, helping in human–human collaboration, and monitoring events and procedures [8]. In practice, researchers were mostly developing software agents capable of learning and adapting to individual preferences.

Towards the turn of the century, *context-aware and ubiquitous computing* emerged as research fields. They adopted the idea of a computer co-operating with the user, but this time also within physical, not only computational environments. Researchers suggested that providing services, such as presentation of information, execution of services, and storage of information [1], should be context-sensitive. Context-awareness has been based on an assumption that the number of potential services can be selectively reduced by collecting extra information about the situation.

A more elaborated view on *how* the user and computer should collaborate was provided by Intel Research in their *Proactive Computing* proposal. There, emphasis is mostly on *process control.* The user is assumed a monitoring role while physical agents take care of the practical actions and lower-level decisions [13].

Despite the references to proactive functionalities in the approaches presented, researchers have not systematically attempted to chart the opportunities of proactivity from the user's point of view.

Characteristics of Proactivity

Starting from a common-sense definition, the concept *proactive* refers to two critical features of a system: 1) that the system is working on behalf of (or *pro*) the user, and 2) is taking initiative autonomously, without user's explicit

command. In other words, *proactive behaviors* are intended to cause changes, rather than just to react to changes.

In a deeper level, if we look at how the concept has been used in the literature, and realized in various prototypes, it can be characterized by the following attributes:

- *Real-time operation*. The system actively follows the on-going activity and the context.
- *World model*. In order to track activity and make inferences, the system must know the basic ontology and dynamics of its context. World model can be explicit (a special representation in system's memory) or implicit (being implemented by the designer directly into system's reactions to changes in the context).
- *Hypothesized goal state*. In order to work "pro" the user, a hypothesis is needed about what user's goals are. That is, there is a state in the world model that the system should help the user to achieve.
- *Sensitivity to future alternatives*. A system is not proactive if it works only in one way in every situation. It must use its real-time information about user's goal to make choices between different future alternatives.
- *Taking initiative*. Based on deliberations above, system undertakes actions to bring user closer to his/her goals.

Even systems with very simple reasoning schemes may contain *some* of these characteristics. Hence, the list above is not a comprehensive set of necessary attributes, but rather a list of family resemblance characteristics. For instance, in the case of a sliding door, that some regard as the best existing example of proactivity, the designer has assumed a goal state: the user wants to get through the door as he or she approaches it. The *implicit* world model includes the user, his or her movement, the door itself, and a capability to slide the door. The door makes its decision real-time, but is not sensitive to future alternatives. Therefore, some might not regard it as proactive. Although not meant to be comprehensive, the above characterization intends to provide the readers a better grasp of the concept. It is also an introduction to our typology of proactive behaviors.

PROACTIVE RESOURCE MANAGEMENT

The main contribution of this paper lies in our proposal of looking at proactivity from a viewpoint of *resource management.* The typology partly arises from our experiences with proactive computing systems and is partly inspired by ethnomethodology, a branch of social sciences that looks at the resources people need in achieving goals in their everyday lives. According to the typology, a system can be advising on, preparing, optimizing, manipulating, inhibiting, or finalizing user's *resources* that can be computational artifacts, processes, or representations (see Table 1 for examples). Hence, the modes are organized temporally, according to the stage at which the system makes the intervention, starting from preparation and ending to finalization. When the system does not behave only reactively, responding to events in the world, but with sensitivity to user's goals and future alternatives, its modes of resource management can be called proactive. In the rest of this section, we present the typology and give examples of the modes.

1. Preparing resources *Examples*: Predictive database searches in the background; a projector that warms itself up before a meeting [12]
2. Optimizing resources *Examples*: Congestion control in Internet traffic; intelligent selector for network services [5]
3. Advising on the use of resources *Examples*: WLAN bandwidth availability advisor [12]; MS Office Assistant; automatic calendar opener for appointments in emails [2]; recommending worth-to-visit web pages [7]; advisor of traffic jams and alternative driving routes [3]
4. Manipulating resources *Examples*: Automatic calendaring of appointments in emails [2]; Wristcare safety monitor[1]; automatic bulleted lists in MS Word; automatic payments in public transportation [3]
5. Inhibiting resources *Examples*: Slide presentation confidentiality screener [12]; in-car ignition interlock device
6. Finalizing resources *Example*: Intelligent hibernation in small devices

Table 1. Modes of proactive resource management.

Preparing New Resources

By "preparing new resources" we mean that the computer takes an active role by initiating new resources (physical, computational, artifacts etc.) for the user. In doing so, it bases its action on an assumption that the new resource is needed or is superior to the existing ones. Resource preparation needs not to be visible to the user.

Considering the examples in Table 1, database searches in the background prepare new information resources that the user may need. If each query takes 1 minute of time before results can be seen, it pays off to develop a model that predicts future queries and launches them in the background on regular basis and caches the results. If the user happens to query something else than was expected, the system can discard the cached results without disrupting the user.

In the case of a video projector automatically warming up, the resource being prepared is the ability to show slides, and the preparation can be based on a prediction from a calendar data and user's movement patterns [12].

Optimizing On-Going Resources

By "optimizing on-going resources" we mean adjusting a resource in use so that the effect of user's actions is maximized. Like in the previous mode, the adjustments are based on assumptions about user's goals.

Both of the examples in Table 1 are from mobile computing domain. If the user is known to exhibit a certain pattern of

Internet use, the system may start rearranging its network configurations to avoid probable congestions. It may also try to subscribe to the optimal service provider with respect to user preferences, quality-of-service, or information needs of the user [5].

Advising on the Use of Resources

By "advising" we mean that the system suggests, highlights, or otherwise emphasizes resources that it expects to be useful to the user, but that the user would not probably notice him/herself. The user is retained the power to actually initialize the resource.

For instance, the automatic calendar opener in a prototype called LookOut exhibits this behavior. LookOut is a mail program that parses incoming mail and tries to infer if appointments are suggested in the text. If it finds such an excerpt, it can open a calendar for the user at the right day or a week, depending on how sure it is of the time that is mentioned in the email [2]. Also the MS Office Assistant with its suggestions provides this mode of proactive assistance.

Manipulating Resources

By "manipulation", we refer to changing the behavior and configuring functional attributes of a resource in a way that resembles the actions user could need to reach the goal. Whereas the previous modes may have been mostly unobtrusive to the user, this mode is something that the user is likely to notice and may not like. Manipulation raises questions of trust and control, which demands responsible thought from the part of the designer.

Probably the most wide-spread example in the list of examples is MS Word's automatic bulleting feature: if a user types a hyphen and a space in the beginning of a line, and presses enter after having written something, Word changes the line into an item in a bulleted list. This feature is based on a simple if–then rule. A more complex example is another feature of LookOut: If LookOut's confidence of understanding the mail message is high enough, it may merely put the appointment automatically into the electronic calendar, and only after that notify the user [2]. A yet another example is the Wristcare safety monitor, which is a product for elderly and outpatients. This watch-like device monitors user's biosignals and is able to notify an emergency center. In this case, the system does not necessarily replace user's actions (since the user may be unconscious), but does something that the user would be doing if he/she could.

Inhibiting the Use of Resources

By "inhibition", we mean deliberate complication of the use of some resource. Reasons for this are the expected eventual future risks to the user. Because of its obvious obtrusiveness, examples of such modes of management are rare.

One pervasive computing scenario mentions a video projector that is able to recognize a budget slide that should not probably be shown to the audience, based on estimates from face recognition and familiarity ratings of attending people. It can then warn the user about a probable mistake or even inhibit its presentation [12]. An existing commercial example is the ignition interlock device that inhibits a drunken person from driving the car. This product blocks the ignition system if person's breathe test is positive, saving him/her and other people from possible injuries.

Proactive Resource Management in Mobile Image Sharing

Lasse is watching a soccer game from a big screen together with a group of close friends in his favorite local pub. When the evening climaxes, Lasse asks a waiter to take a few pictures of him and his friends with the big screen in the background. Lasse wants to send the pictures to Kalle, his workmate who is also watching the game. He gives a short description for the group of images ("Sweden–Portugal 2–0!") and uploads them to his server for Kalle to view. One totally black image is removed by the proactive system from the batch *(4)*, as exposure settings for that image were not suitable for a dark pub. The system believes that in this setting, a pub, a dark picture represents a failure in the automatic imaging setup rather than anything else.

As many ISP options are available in the pub, each with different prices and connection speeds, and these are changing in time, the proactive system helps Lasse by selecting the one with the fastest connection rate available at that moment *(2)* (see [5]). This assumption is based on a hypothesis that images taken in social situations (many faces in the image) are more often *immediately* shared with others, in contrast to non-social images that are more often related to personal memorabilia and thus only archived (see [4]).

However, Lasse's frantic photographing has consumed camera's batteries almost to minimum. Since Lasse is sitting downstairs where the network strength is weak, sending shots from the camera would very likely consume all the remaining power. Expecting that Lasse would like to keep the device available also for phone calls, the phone does not let Lasse send all the pictures until he is back in the street level or other place where the field is stronger *(5)*.

Because the image sharing server organizes the pictures according to metadata such as location and people, mobile images have to be annotated. Annotation, however, takes time and can thus interrupt the on-going activities. Therefore, the system does not force Lasse to annotate the image immediately, but proactively suggests *(3)* doing so later on after he has left the pub and is walking home. As Lasse thinks that he does not want to do that while walking, and does not respond to the prompt, the system finishes up *(6)* the annotation task by providing best guesses to the metadata fields. It fetches location information from a positioning service and names of the people by recognizing faces and mapping them to previously annotated pictures in Lasse's image server (see [11]).

For Kalle, with whom Lasse wanted to share the pictures, the system starts the automatic download as soon as it recognizes that immediate sharing might be important for this kind of picture batch. The pictures, not only thumbnails, are therefore prefetched to Kalle's mobile phone's cache and are ready to be accessed as soon as Kalle wants to *(1)*.

Figure 1. A mobile imaging scenario illustrating the typology. The cursive numbers in parentheses refer to the modes in the typology presented in Table 1.

Finalizing Resources

By "finalization" we mean operations that terminate a resource. If the user wants to have the resource available again, he or she must invoke it again.

Examples of finalization are automatic form-filling based on user profiles and intelligent turn-off/hibernation functions in small devices that try to save battery life.

DISCUSSION

Proactive or context-sensitive?

We now return to discuss briefly the differences between proactive functionalities and contextually triggered actions in general, in the light of the characteristics presented.

Proactive functionalities appear to be a subset of context-triggered actions, their distinctive feature mostly being deliberation of actions with respect to user's goal state in the situation, not any level that is optimal to the system itself. The difference can be subtle, however. In the case of the ISP service provider selector (see Table 1) that optimizes user's wireless resources, the difference comes from how the system conforms to user model that governs its activity. We believe that mere automatic configuration to the context is not proactive behavior, according to the characteristics.

Innovating New Uses for Proactive Systems: A scenario

One of the main motivations for constructing this typology was to innovate new possibilities for proactive actions in our applications. The mobile imaging scenario in Figure 1 illustrates how the typology, by revealing new possibilities in how to manage users' resources, can spring new ideas for uses of proactivity in non-trivial application domains.

The scarcity of proactive consumer products, compared to the amount of research interest over the years, suggests that conceptual reflections on proactivity are motivated. The scenario provided in Figure 1 attempts to prove the benefit of this kind of conceptualization. With a simple exercise of inventing one proactive functionality from each mode revealed possibilities that have not yet been considered or done in the existing mobile imaging systems.

CONCLUSIONS

Thus far, proactive computing has not fared very well. There are only few if any successful consumer products, and overall, the hit rate of R&D has been very low, which has led to pessimistic views about the whole enterprise (see [9]). Almost every major advancement in HCI has been preceded by conceptual developments that have assisted researchers to better grasp their problems. Needless to say, but proactive computing is still lacking its own concepts and terminology. In this paper, we have proposed a user-centered typology for proactive behaviors that, we hope, may help researchers to see proactive behaviors more from the user's point of view than before. We have placed the typology in its historical and analytical context. Moreover, we have evaluated the typology by showing that it can inspire designers to innovate new ideas of how proactivity may take its place in an application.

This paper has shown that proactive actions need not necessarily replace user's actions, which seems to be the most often encountered interpretation of proactivity in HCI at the moment. As we have argued, user's resource management can be supported in many different ways through proactive assistance. With the suggested typology, designers can better keep in mind the breadth of types of assistance they can build into their proactive systems.

ACKNOWLEDGMENTS

We thank Sauli Tiitta, Miikka Miettinen, Matti Rantanen, and Martti Mäntylä for comments on the typology. AO acknowledges funding by the Academy of Finland.

REFERENCES

1. Dey, A.K., Abowd, G.D. and Salber, D. A conceptual framework for supporting the rapid prototyping of context–aware applications. *Human–Computer Interaction 16*, 2–4 (2001), 97–166.
2. Horvitz, E. Principles of mixed-initiative user interfaces. *Proc. CHI 1999*, ACM Press (1999), 159–166.
3. ISTAG (2001). Scenarios for ambient intelligence in 2010. Information Society Technology Advisory Group, European Union, Sixth Framework Programme.
4. Koskinen, I., Kurvinen, E., and Lehtonen, T-K. *Mobile Image*. IT Press, Helsinki, Finland (2002).
5. Lee, G., Faratin, P., Bauer, S. and Wroclawski, J. A user-guided cognitive agent for network service selection in pervasive computing environments. *Proc. PerCom 2004*, IEEE Computer Society (2004), 219-228.
6. Licklider, J.C.R. Man–computer symbiosis. *IRE Transactions on Human Factors in Electronics*, HFE-1 (1960), 4–11.
7. Lieberman, H. Autonomous interface agents. *Proc. CHI 1997*, ACM Press (1997), 67–74.
8. Maes, P. Agents that reduce work and information overload. *Communications of the ACM 37*, 7 (1994), 31–40 and 146.
9. Oulasvirta, A. Finding new uses for context-aware technologies: The humanistic research strategy. *Proc. CHI 2004*, ACM Press (2004), 247-254.
10. Rich, C. and Sidner, C.L. Adding a collaborative agent to graphical user interfaces. *Proc. UIST 1996*, ACM Press (1996), 21–30.
11. Sarvas, R., Herrarte, E., Wilhelm, A., and Davis., M. Metadata creation system for mobile images. *Proc. MobiSys 2004*, Boston, MA, USA. ACM Press.
12. Satyanarayanan, M. Pervasive computing: Vision and challenges. *IEEE Personal Technologies 8*, 4 (2001), 10–17.
13. Tennenhouse, D. Proactive computing. *Communications of the ACM 43*, 5 (2000), 43–50.