Soft Actuation for Home and Office

Jarosław Domaszewicz
Institute of Telecommunications
Warsaw University of Technology
Warsaw, Poland
domaszew@tele.pw.edu.pl

Spyros Lalis
University of Thessaly
& IRETETH/CERTH
Volos, Greece
lalis@inf.uth.gr

***Abstract*—Nowadays, most effort in the area of context-aware systems goes into applications that process sensor data to proactively drive actuators. We share the concerns raised about such fully automated operation. Most notably, due to imperfect context inferences, actuating decisions are often contrary to the user's actual desires. Thus we focus on what we refer to as soft actuation: issuing low-key, non-verbal hints to the user, prompting him to optionally perform specific actuating actions. An actuating action consists in reaching to a nearby object and performing a simple manual operation on it. In this paper we describe the concept of soft actuation, position it with respect to related work, and identify relevant research challenges.**

Keywords—context-aware applications; pervasive computing; human-computer interaction; embedded interaction; user control; intrusiveness; home and office environment

I. INTRODUCTION

The prevalent vision of pervasive computing is that applications should operate *proactively*, i.e., (a) sense, (b) infer higher-level context, (c) decide how to affect the environment, and (d) actuate accordingly. The last step of this sense-and-react chain, actuation, is executed by the application itself. It is usually an operation on some object, e.g., switching on a light, rolling up blinds, or turning down a thermostat. This is exemplified in Fig. 1a, for a simple wintertime application that protects the user from catching a cold, by closing the window when the inside temperature drops below a threshold.

While such a proactive approach may appear to offer most added value (as the user does not have to do anything), it has serious pitfalls, indentified in the literature. For example, Bellotti and Edwards [1] stress the difficulties of treating people as "contextual entities," whose states could be faithfully modeled and inferred by the system. As a result, proactive actuation often turns out to be contrary to the user's actual desires. They conclude that "human initiative is frequently required to determine what to do next." Similar arguments are put forward by Intille [2], who envisions a sensor-instrumented home, where context-aware applications do not actuate proactively, but subtly hint the user to perform the actuation himself. The final decision is left to the user, who may not pay attention to the hint because he is too focused on the task at hand, or decide not to perform the suggested action because it is contrary to his current preferences. The system becomes naturally resilient to wrong context inferences and offers a high level of user control. A similar philosophy is advocated by Streitz et al. in [3], where "system-oriented, importunate smartness," consisting in proactive operation of a smart space, without a human in the loop, is contrasted with "people-oriented, empowering smartness", which consists in making suggestions to the user, who "can always decide what to do next." More recently, pitfalls in proactive, fully automated operation, as well as possible remedies, are presented in [4].

Indeed, let us consider the sense-and-react loop shown in Fig. 1a. There may be a number of reasons why proactive actuation may not be well received by the user. First, there may be "objective" errors in context sensing and inferencing, which may lead to obviously unreasonable actions ("Something must be wrong! Actually, it's quite warm inside, yet the window has been closed."). Second, even if there are no such errors, the user may occasionally desire something opposite to what the system does ("Yes – it's very cold, but I still want some more fresh air."). Third, even if proactive actuating actions are perfectly aligned with the user's desires, the action can be unexpected and distractive ("It's OK that the window has been closed; it's just disturbing when this suddenly happens by itself."). Finally, even if none of the above hold, the user may feel a (possibly vague) sense of lack of control ("The window closing function is useful and non-distractive, but I feel uneasy that the system makes decisions about my environment."). In Fig. 1a we summarize these potential problems by depicting the user as being puzzled or even upset.

In this paper we explore an alternative to proactive actuation, along the lines of arguments and visions presented in [1-4]. We focus on context-based, application-generated hints that suggest the user to do specific actuating actions. An actuating action typically consists in reaching to a nearby object and performing a simple manual operation on it. Accordingly, a hint specifies both the object to be acted upon and the operation to be performed (e.g., "close the window"). We call the delivery of hints *soft actuation*, since the purpose of a hint is to trigger actuation, but the decision as well as the execution (if any) is left to the human. The optional actuating action of the user, who becomes a part of the control loop, is referred to as *hard actuation*.

Allowing the user to decide makes soft actuation unreliable. For any given hint, the application does not know whether the hard actuation *will* occur, and it may not have a direct way to know whether it *has* occurred. Thus soft-actuating applications could also be called "best-effort" sense-and-react applications. Clearly, the functionality of such applications should not be life-critical – all hints should be of low importance and low urgency.

978-0-7695-5038-1/13 $26.00 © 2013 IEEE
DOI 10.1109/IE.2013.32

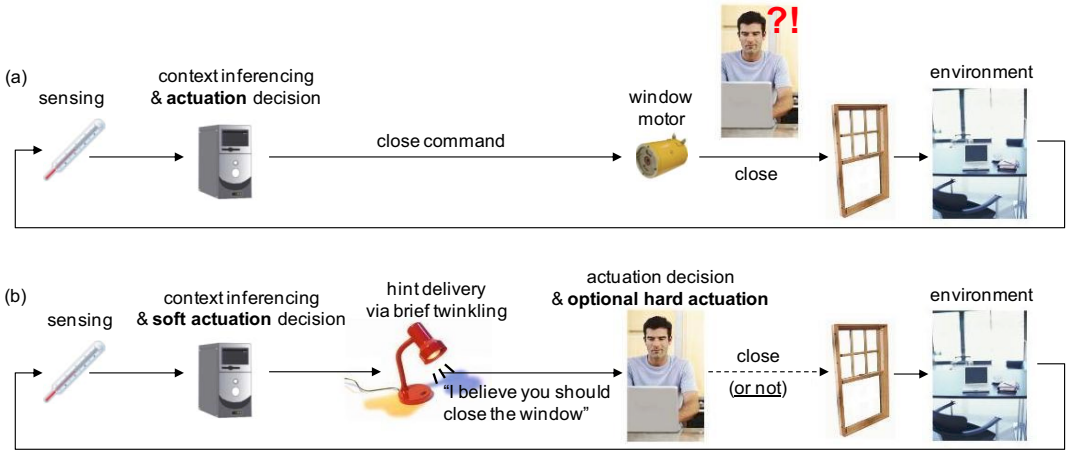

Figure 1. Sense-and-react loop with (a) pro-active actuation (without a human) and (b) soft actuation (with a human in the loop).

Our focus in this paper is on soft actuation for the *regular user in the home or office*, as opposed to the experienced or professionally trained user in a specialized environment (e.g., the pilot in the cockpit). We confine ourselves to *non-critical* applications. We pay special attention to the requirement that soft actuation should be *non-intrusive*. We cover the interaction technique in its entirety, i.e., the whole involvement of the human in the loop, including the decision to actuate and the actuating action, not just the delivery of hints. On the software side, we touch on some emerging middleware-level issues, but leave out application programming.

In this paper we make the following contributions. First, we systematically describe the concept of soft actuation for home and office; as indicated above, the high-level idea has been put forward quite some time ago, but a systematic treatment is missing (to the best of our knowledge). Second, we position soft actuation with respect to related research areas. Finally, we suggest that soft actuation becomes the subject of a systematic study, identify selected research challenges to be addressed, and offer a few preliminary ideas as to potential solutions. The discussion is kept at the conceptual level; an experimental validation with real users is the subject of ongoing work.

This paper is structured as follows. In Section II, we include the systematic description of soft actuation. In Section III, we offer an extended example of potential use of soft actuation in the office, by listing a number of hints that could be issued there. In Section IV, we describe related work, while in Section V we identify the research challenges. Finally, in Section VI, we briefly comment on an ongoing validation experiment and conclude the paper.

II. Concept of Soft Actuation

Proactive and soft actuation are contrasted in Fig. 1. The figure also presents a possible way to achieve soft actuation – namely, with a regular, low-tech object; the lamp next to the user twinkles briefly to hint that he should close the window. This exemplifies our emphasis on hint delivery done in a low-key, *non-verbal*, *embedded* way. In particular, to keep hints subtle (and easy to ignore), they should *not* require the user to interact with an attention-grabbing device, such as a mobile, tablet, or PC ("no screens"). Actually, in the spirit of calm technology [5], we treat the non-intrusiveness of hints equally central to the concept of *soft* actuation as the fact that the application does not do hard actuation.

From the architectural point of view, our concept of soft actuation affords a number of "degrees of freedom," not all exemplified in Fig. 1b. These are as follows. (i) A contextual condition that triggers hint delivery may be based on data sensed from multiple different sensors or objects. (ii) The context-providing object(s) may be unrelated to the hint-delivering object, which in turn may be unrelated to the object to be acted upon (called the *target object*). (iii) A soft-actuating application may generate different kinds of hints, each corresponding to a different contextual condition and pointing to a different operation and target object. For example, in Fig. 1b, different hints would be conveyed with different twinkling patterns. (iv) A pervasive computing platform may allow concurrent execution of independent soft-actuating applications, each producing its own hints. And finally, (v) multiple hint-delivering objects may be used.

Even for most complex combinations allowed by the above, the set of all hints that a soft-actuating platform can produce is pre-defined and fixed (it changes only when a new application is added). This makes it possible to hint without words, using non-verbal signs (e.g., twinkling patterns) that the user should eventually learn to recognize instantly.

In Fig. 2 we present a minimalistic model of hint reception by the user. For the sake of simplicity we sequentially order the "steps" of the reception process. As shown, hint reception may end with different outcomes. Given the required non-intrusiveness, a hint may remain simply *undetected* (A). If detected, the hint may not be paid attention to, probably because the user's task at hand is of sufficiently high priority; in this case we say that the hint is *ignored* (B). If the user does decide to pay attention, the hint may still remain *unrecognized* (C); hint recognition may get difficult if too many different

hints are possible. If the hint is recognized (understood), the user decides whether to do the suggested hard actuation. He may choose to do nothing; in that case we say that the hint has been *rejected* (D). Otherwise, the user performs hard actuation on the target object, in which case we say that the hint has been *accepted* (E). In the latter case, the actuating action need not be performed immediately; for example, the user may proceed to close the window a few minutes after receiving a respective hint, when it is more convenient to do so.

We assume that the user does not provide any input to the soft-actuating system (e.g., to confirm one of the B through E outcomes); the need to provide such input would contribute to the overall intrusiveness. The only action performed by the user is the actuating action itself. The application can detect the outcome only by sensing the target object (if possible), or indirectly, by context inferencing. Thus soft actuation is a one-way interaction technique.

Note that we make a distinction between a hint being ignored (B) and rejected (D). The former occurs when the user does not wish to pay any attention to the hint – not even to recognize it. Importantly, for the sake of low intrusiveness, a hint should be delivered so that it is easy to be ignored. The latter (D) occurs when the user has recognized the hint, but does not act on it.

The hard actuation decision, i.e., to accept (E) vs. reject (D), may be affected by whether the user understands *why* a hint has been delivered. As mentioned above, the delivery of a hint is triggered when the application detects a certain contextual condition. We call a brief, natural language description of the respective condition the hint's *rationale*. For example, a rationale for the "close the window" hint may be "the room is too cold." A given hint, which, in our approach, specifies only the target object and an operation on the object, may have more than one rationale. For example, there may be other reasons to close the window. Thus it may be meaningful to extend our hint reception model and allow the user to optionally retrieve the rationale when a hint is delivered (we comment more on this in Section V.A, under "hint intelligibility"). Over time, however, the user may learn to use his own judgment or simply trust the system most of the time, without an explicit, system-provided rationale.

Rationales aside, the user may reject a hint (D) for at least two reasons. First, the suggested action may be simply impossible to perform (e.g., a hint to close a window that is already closed), or it may clearly not make sense (e.g., a hint to close the office door of a noisy office to avoid disturbing others, even though the office is perfectly quiet at the moment). We call such a hint *puzzling*. Issuing puzzling hints has to do with insufficient sensor instrumentation or too crude context inferences about the physical environment. Second, even if a hint is not puzzling, i.e., makes perfect sense in the current state of the environment, the suggested action may be contrary to the user's actual desires (e.g., a hint to close the window to avoid catching a cold, when the user badly needs more fresh air). We call such a hint *undesirable*. Issuing undesirable hints has a more fundamental reason: the inability to properly model a human with his changing preferences, emotional states, etc. Note that both of the above problems are far more pronounced

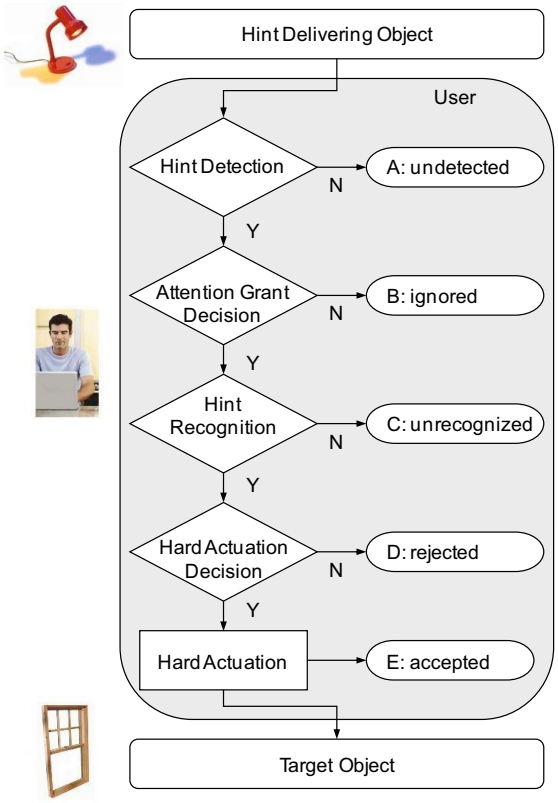

Figure 2. Hint reception process.

with proactive actuation: there, the user faces not a low-key hint (which he may reject), but the application-executed actuating action itself (as well as its possible consequences).

It is important that the user clearly understands the "nature" of the hint-based interaction. First, the user must be prepared to receive some puzzling and undesirable hints. Second, the user must know that any hint reception outcome (from A to E) is perfectly legitimate and acceptable; in particular, he should not worry about not detecting a hint, and he should not feel obliged or pressured to recognize or accept a hint. Ignoring and rejecting are just as good as recognizing and accepting, respectively. Even a failure to recognize a hint (C) should not be treated as a problem (although one would not like this to occur too often). These are the key principles of our interaction concept, made possible by considering only non-critical applications. Incidentally, soft-actuating applications should be programmed as best-effort ones (i.e., hard actuation cannot be taken for granted), and the application logic should tolerate any hint reception outcome.

In our opinion, the overarching issue in soft actuation is its *efficiency-intrusiveness tradeoff*. Efficiency has to do with detecting and recognizing hints; one could tentatively define it as the product of two ratios: $(N_{\text{detected}} / N_{\text{delivered}})$ and $(N_{\text{recognized}} / N_{\text{non-ignored}})$, which capture hint detection efficiency and hint recognition efficiency, respectively. We do not attempt to define intrusiveness here, but consider low intrusiveness of soft actuation essential. A major factor for both dimensions is how hints are designed and delivered. It is easy

TABLE II. INDICATIVE HINTS IN THE OFFICE

Hint		Hint Rationales		Functionality Cluster
Object	*Operation*			
Lights	Switch Off	R1	There is enough sunlight to illuminate the room.	Lighting (energy saving, ergonomics)
		R2	Lights should be switched off for the night, before leaving.	
	Switch On	R3	It is considered unhealthy to work in the dark.	
Thermostat	Turn Down	R4	The room is too warm.	Heating & Ventilation (energy saving, comfort)
		R5	The window is now open, avoid wasteful heating.	
		R6	Heating should be turned down for the night, before leaving.	
	Turn Up	R7	The room is cold.	
		R8	The window has been closed, heating can be turned on again.	
Window	Close	R9	The room is getting cold.	
		R10	The window should be closed for the night, before leaving.	
	Open	R11	The window has been closed for long, let some fresh air in.	
Door	Close	R12	The window is open, no need to reduce temperature in the halls.	
		R13	You are having a phone/Skype call, you may want to keep it private.	Social Aspects (office policy, privacy, etc.)
		R14	It is too loud in the room, avoid bothering your colleagues.	
	Open	R15	The door has been closed for a long time, respect open door policy.	

to produce hints that are non-intrusive but hard to detect and recognize, as well as ones that are immediately recognizable but highly intrusive. Another factor is the making of a hard actuation decision: it could noticeably contribute to the overall intrusiveness, probably more so than the hard actuation (actuating action) itself. In fact, in the office environment, performing the actuating action, which usually requires some physical activity and not much thinking (e.g., getting up and closing the window) may be considered a desirable short break from sedentary work.

Overall, we hypothesize that a distinguishing feature of soft actuation is that the entire hint reception process (Fig. 2) is *inherently simple and non-demanding* and can be made non-intrusive. If so, soft actuation would be a way to provide the benefits of automatic (application-driven) context sensing and inferencing, while ensuring that the user retains a high level of control over his environment.

III. EXAMPLE: SOFT ACTUATION IN THE OFFICE

Imagine an office equipped with a modest sensing infrastructure, as described in Table I. The sensors are placed at different locations in a targeted way, e.g., a temperature sensor is attached near the window to detect if it is opened or closed, by observing the local temperature variations.

TABLE I. SENSING INFRASTRUCTURE IN THE OFFICE

Sensor	Location	Measurement/Detection Purpose
Temperature	Desk	Room temperature
	Window	Window opening/closing
	Radiator	Thermostat setting
Brightness	Desk	Room brightness
	Window	Brightness due to sunlight
Motion	Desk	User presence
Noise	Desk	User activity
Magnetic	Door	Door opening/closing
Software	PC/phone	Phone/Skype call

Table II lists indicative hints that can be produced by a soft-actuating context-aware application, which uses the sensors from Table I. The hints are listed along with their rationales and clustered in distinct areas of functionality. We have checked the feasibility of the hints and rationales (given the available sensors), by developing application logic for the respective contextual conditions. For some hints we assumed wintertime, with outside temperatures around or below zero degrees centigrade. Clearly, a substantial number of hints can be generated for different target objects. Some hints have multiple rationales. The respective actuating actions are simple and natural to perform. The system can contribute to worthy goals: energy savings, good working conditions, and considerate behavior towards colleagues.

The contents of Table II give rise to a discussion of reasons for adopting soft actuation. Consider the rationales R5 and R8. The goal is to avoid heating when very cold air enters an office through an open window: the thermostat should be turned down when the window is opened, and then up again when the window is closed. Jointly, they are analogous to the "open window function" found in advanced thermostats [6]. It is quite unlikely that the user would find proactive actuation by such a thermostat objectionable. Thus, in this case the main reason for adopting soft actuation is simply the lack of respective actuators. This is significant, as most offices do not feature electrically operated windows, doors, and blinds, presence-sensitive lighting systems, or latest-generation thermostats; retrofitting them with such features would be prohibitively expensive. With soft actuation, the problem of the lack of actuators disappears by definition.

Some rationales (e.g., R1, R4, R7 and R9) exemplify other reasons, ones most often raised in the literature: imperfect sensing and context inferencing, failures to determine the user's actual desires, distraction resulting from an unexpected actuating action, and a general sense of lack of control. Soft actuation seems to address all of these concerns, at least to a degree. For example, the technique appears inherently more tolerant to sensing and inferencing errors, and thus can be implemented with fewer sensors and simpler application logic. Puzzling and undesirable hints can simply be rejected.

As an aside, consider the case of offices occupied by multiple users. There, the above problems with proactive actuation are aggravated, as it is more likely that at least one

person will find a proactively executed actuating action objectionable. With soft actuation, a hint may be briefly discussed by present occupants, and the decision to accept or reject may represent a consensus.

Finally, consider the door hints, with rationales R12-15 (an open door policy, common at universities, is assumed). They uncover yet another case for soft actuation: certain actuating actions should not be performed proactively for very basic psychological reasons. One can try to imagine the sense of imprisonment or a total lack of privacy if the office door would close or open on its own. A respective proactive system would likely be considered "arrogant" and "disrespectful." For such cases, proactive actuation is not an option, but soft actuation seems to fit perfectly: the system needs to issue a gentle hint to the user, just like a friendly person would. To summarize the above discussion, we depict the range of problems with proactive actuation in Fig. 3.

IV. Related Work

One should start by noting that soft actuation is already used, albeit in a limited way, in a number of familiar home products. Consider the simple, mostly non-intrusive beeps produced by a washing machine ("washing cycle is over, remove laundry"), or a coffee machine ("coffee is ready, pick it up"). In such cases, the *same* object provides context data, delivers the hint, and acts as the target object. We decouple these roles, and thus our concept is more general. Still, we find such proven examples of hint-based interaction encouraging.

The most relevant real-life example known to us, almost perfectly matching our description of soft actuation, is the window signaling system in an office building [7]. Such systems are deployed in mixed-mode buildings, i.e., ones that include both an air conditioning system and human-operable windows. The latter are meant to increase the occupant's sense of personal control, but also to allow natural ventilation in a cost-effective way. A window signaling system informs the occupant about preferred time slots for opening a window. These time slots are derived from indoor and outdoor sensor data (primarily the temperature, but also humidity, wind speed, and CO2), taking into account both comfort and energy efficiency. The "hint delivering object" typically consists of a panel with two lights: green for "open" and red for "close."

To be precise, one should observe that in our formulation a hint is meant to trigger the actuating action (roughly at the time of the hint's delivery), while, in a window signaling system, the window is meant to be opened (if at all) at *any* time during an "open" slot. Still, that is a minor difference, and similarities abound. For example, the user does not have to follow a window signaling system's suggestion. The system is not life-critical; even more so, "window use transgressions don't pose any serious performance risks" [7].

While the described window signaling bears strong resemblance to soft actuation, the former can only be considered a special, highly-targeted case of the latter. In a window signaling system there is only one kind of target object (i.e., windows) – that is why hint delivery can be so simple. In our more general formulation, hints may refer to *multiple* target objects, each affording different operations.

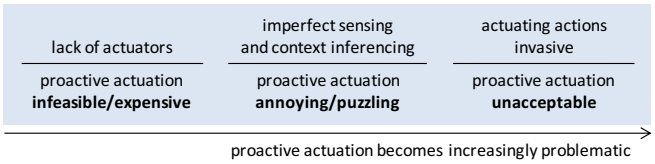

Figure 3. Problems with proactive actuation.

Interestingly, as pointed out in [7], the window signaling systems require further research. Currently, they seem to affect the behavior of only a minority of occupants; many users reported a tendency not to pay attention to the signals. This may have to do with the method of the hint delivery, which is chosen in an ad-hoc way. As the authors say, "there was little systematic discussion about the design of the signaling device."

One of the earliest examples and arguments for a soft actuation-like system apparently comes from a social psychology study on energy conservation [8]. There, a light blinks until the user shuts off the air conditioner (when it makes sense to do so). The authors claim that such "systems focus people's attention on specific conservation actions and do so exactly when these actions are appropriate." Their experimental "hint," however, is extremely intrusive. For example, as to a lamp, we envision brief, gentle twinkling, rather than prolonged blinking.

In pervasive computing, the basic idea of soft actuation was also put forward quite some time ago. As said, the rationale is convincingly presented in [1-4]. In [2], hint delivery is also envisioned: a LED in a window's frame blinks when *that very* window should be opened or closed (i.e., a hint is delivered by the target object). Yet it is probably unrealistic to assume that each object is individually enhanced for hint signaling. Incidental implementations of soft actuation can be found, e.g., in [9] where a blinking lamp hints the user to water a plant. However, to the best of our knowledge, soft actuation in the home or office has not yet been systematically investigated.

Soft actuation might be seen as an example of passive context-awareness, as defined in [10]: the application uses a hint to present a contextual condition (corresponding to the rationale) to the user, without taking action. However, such a view would not fully capture the hint "semantics"; the essence of a soft actuation hint is to invite to a specific actuating action, not to inform about context.

Ambient (peripheral) information systems [11] offer a non-intrusive, non-verbal way of presenting information, without using attention-grabbing screen devices. In that sense, a hint-delivering object is (a kind of) an ambient display. However, the emphasis in ambient displays is to present the current value of a continuously varying quantity (e.g., temperature). In contrast, soft actuation hints, issued when a context-aware application decides that some actuation is desirable, are more like assorted, irregularly occurring events. Thus the structure and semantics of presented information is different. Still, techniques and approaches developed for ambient information systems will likely be useful in soft actuation.

It might be argued that some reminder and notification systems implement soft actuation, e.g., see [12] for a system

that helps the user to follow a medication program. One important difference is that the goal of a reminder is to make the user recall a scheduled task that should not be missed, whereas a hint points to an *unscheduled* and *optional* action (and can be ignored or rejected). Unlike most message notifications, hints come from a pre-defined set and can be communicated non-verbally. Further, the entire hint reception process seems to be decisively simpler than receiving and acting on an arbitrary textual message.

Soft actuation may appear no different from persuasive technologies [13]: in both cases the point is to affect human behavior. However, persuasive technologies aim at rather complex behavioral patterns, having to do with habits and important personal goals, e.g., a reduction of TV viewing. To be successful, they appeal to the user's motivation and emotions. The goals of soft actuation are far more humble; hinted behaviors, while likely to prove beneficial to the user, are not meant to contribute to a habit or a longer-term personal priority. Besides being motivationally neutral, they are also very easy to perform. Persuasive systems are often defined as "an interactive technology that changes a person's attitudes or behaviors" [13]. Then soft actuation could be described as *a non-intrusive technology that hints at simple behaviors affecting nearby objects*.

A vision of people acting as actuators, but in the smart city environment, is put forward in [14]. The authors consider city-scale closed-loop applications, and declare that "the inhabitants of the cities themselves can be considered possible agents of regulation and actuation." Yet, it is the home that offers countless natural opportunities for dweller-performed actuation, probably to a greater extent than the city.

As to research in Human-Computer Interaction (HCI), the subject most relevant to soft actuation seems to be human interruption. For example, [15] is a rich source of insight applicable to soft actuation. In particular, our hint reception process might be considered a simplified and specialized version of the information management stage model (IMSM) [15]. We believe that it is productive to consider the delivery of a hint as a kind of interruption, but with a number of distinguishing properties: the domain of regular users at home or office, the low priority of the interruption (with the option to legitimately ignore or reject), the emphasis on non-intrusiveness, shortness and simplicity of the actuating action, and dissimilarity between the user's primary task and the actuating action.

Finally, if the entire digital control loop (like the ones shown in Fig. 1) is assumed to belong to the area of computing, then the concept of soft actuation fits the description of *human computation* provided in [16]. There, a key feature is that "the human participation is directed by the computational system or process." In our case, a "human actuator" is directed by hints issued by a context-aware application acting as the controller.

V. Research Challenges

We have identified a number of issues that need to be researched in order to assess soft actuation in the home or office. Most of them are motivated by the efficiency-intrusiveness tradeoff, and are related to one another. Still, we categorize them into (a) hint design issues, (b) UI extensions, (c) system-level issues, and (d) control-theoretic issues.

A. Hint design issues

Hint modality. Should hints be delivered visually, aurally, or with a mix of the two modalities? The answer probably depends on whether a hint is meant to reach a specific person (e.g., a specific office worker) or any member of a group (e.g., any family member at home). An additional factor is whether the user freely moves around or is expected to occupy a designated place. Accordingly, auditory hints are probably a better choice at home, where people move, and any family member can do hard actuation. This also holds for shared office spaces, if hints can be acted on by anybody. If they are meant to be "personal", then visual ones, delivered on the desk, are probably preferable. Multi-modal hint delivery could also be considered (we provide a simple example below).

Concept of hint-delivering object. As noted, the hint-delivering object should be a kind of an ambient (peripheral) display, not a typical "screen-based" device. Further, we consider verbal hints (e.g., pop-up messages or speech utterances) and musical excerpts too intrusive. Instead, we assume that hints should be delivered via abstract *patterns*, each hint encoded with its own, unique pattern. Some indicative pattern-producing objects are as follows. (a) An AmbientOrb-like object [17] that delivers a hint by taking on the color specific to the hint. Between hints, the orb is grey ("color-less"). (b) A small panel with an array of LEDs. A hint is delivered by displaying its unique pattern of glowing LEDs. Between hints, no diode glows. Such an object could be considered a miniaturized version of the Hello.Wall interface [3]. Optionally, the LED panel could be enhanced with a buzzer, which produces a gentle beep when a LED pattern starts to be displayed. (c) A lamp-like object, which delivers hints with short blinking (twinkling) patterns. Between hints, the object does not emit any light. (d) Any object with a loudspeaker. A hint is delivered via a unique audio pattern (an earcon [18]).

Is it feasible to use regular objects (e.g., a lamp) to deliver hints? What should be the additional features (beyond the ability to produce patterns) of a dedicated hint-delivering object? For example, equipping such an object with a button would allow "hint on demand" (see below).

Hint delivery notification level. What is the optimal hint delivery notification level (to use a term from ambient displays)? In [19], six notification levels are identified: "ignore," "change blind," "make aware," "interrupt," and "demand attention." Referring to our prototypical hint-delivering objects, the orb and the basic LED panel allow the "make aware" notification level, while the buzzer-enhanced LED panel, the twinkling lamp, and the earcon generator seem to impose the "interrupt" level.

From the intrusiveness point of view, the lower the notification level the better – thus "make aware" is better than "interrupt." On the other hand, as reported in [7] for window signaling systems that utilize the "make aware" notification level, quite a few users declared a tendency not pay attention to the signals (hints). Thus the "interrupt" level may be a better

choice. Using a remote analogy, one could argue that in fact a hint can be likened to a CPU interrupt: both are meant to trigger a brief pause in the main task, an actuating action and an interrupt service routine (ISR), respectively.

Hint pattern design. For the hint-delivering objects discussed previously, the patterns would take the form of (a) a single color, (b) a LED pattern (along the lines presented in [3]), possibly preceded by a buzz, (c) a twinkling pattern, and (d) a sound pattern, respectively. How to choose the patterns so that the user is able to recognize them easily? Consider the twinkling lamp. To ease reception, patterns could be structured, e.g., start with a common "header," followed by a hint-specific "body." The header would allow the user to quickly make the attention grant decision. The body could be in turn structured into the operation and the target object. Thus the whole pattern would be a triple: *<header, operation, object>*. Fig. 4 illustrates the concept, via two indicative hint examples.

Moreover, the header could be made hint-specific. For the LED panel with a buzzer, the initial beeps could encode the respective rationale's functionality cluster (see Table II). The sounds would convey the general area of the hint and thus help the user decide whether to look at the LED pattern. This would make the hint patterns truly multi-modal.

Hint reception capacity. What is the maximum number of different hints (patterns) that the user can *comfortably* differentiate and recognize? Note that the answer to this problem depends on the hint modality, pattern design, and the allowed level of intrusiveness.

Hint learning. How should the user learn to associate hint patterns with respective actuating actions? A simple, low-tech solution would be to provide him with a reference page serving as a "dictionary"; the hope is that the user would use the reference page to look up patterns only initially. Another approach is to allow optional "hint inspection" (see below).

Hint intelligibility. As stressed in [1], context-aware systems should be intelligible, i.e., the users should be able to understand their behavior. In soft actuation, intelligibility means that a user receiving a hint should not only be able to tell what operation to perform and on which object, but also to understand (even if in simplified terms) *why* the hint was generated in the first place.

As shown in Table II, a hint can be associated with one or more rationales. We envision three rationale-based techniques of promoting intelligibility of a soft-actuating system. A minimalistic approach is to have the reference page with all (usually just a few) possible rationales for each hint (as in Table II). Alternatively, the rationale for a specific hint could be made explicitly available to the user when the hint is delivered. The rationale could be provided (a) by encoding and appending it to the hint pattern itself (e.g., the twinkling pattern would take the form *<header, operation, object, rationale>*) or (b) by using hint inspection (see below).

Hint replay policy. A washing machine repeats its hint a number of times before giving up. What should a soft-actuating system do? One reason to repeat a hint is to give the user one more chance to become aware of it and/or to recognize it.

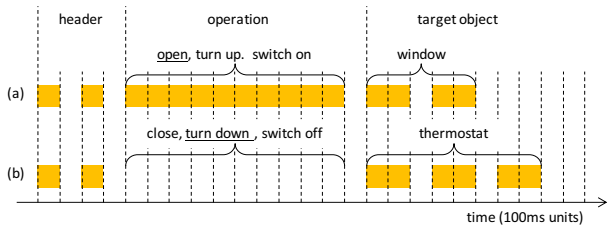

Figure 4. Pattern structure for a twinkling lamp and indicative hints: (a) open the window; (b) turn down the thermostat.

Another reason is that the respective contextual condition is still satisfied. The latter may mean, however, that the user has rejected the hint; in that case the repetition would be highly intrusive. A different approach is *hint on demand*: the user requests a replay of, say, the most recent hint that is still valid.

Intrusiveness level. How to evaluate intrusiveness associated with different outcomes of the hint reception process? Assuming that the hint delivery is carefully crafted, can the whole process be made reasonably non-intrusive (as we hypothesize)? In our concept of soft actuation, low intrusiveness is a key objective; it affects a number of design decisions mentioned above.

B. *UI extensions*

Hint inspection. While we insist on embedded, non-verbal hint delivery, it could be, at the user's discretion, extended with some *conventional* user interaction. For example, a mobile device could be used for *hint inspection*: after "scanning" a visually delivered hint pattern with the device's camera, its meaning is displayed as text. This could prove useful for hint learning. For the sake of intelligibility, the device could also display the hint's rationale. A similar approach is presented in [3], where a personal device called ViewPort can display a textual counterpart of an abstract pattern shown by the Hello.Wall ambient display.

C. *System-level issues*

Centralized vs. distributed hint delivery. How many hint-delivering objects should there be? One solution is to use a single object for all hints the system can generate. Alternatively, hints could be delivered via different objects, depending on the target object, the user's location, or the application generating the hint. This is an example of a more general issue of context-aware hint delivery.

Hint-related middleware services. In an open pervasive computing platform, hints can be generated by independently developed and concurrently running applications. How to support that at the platform level (the platform's hint-oriented API, assigning patterns to applications, etc.)?

An intrusiveness-constrained multi-application system. How to control overall hint intrusiveness when hints are generated by multiple applications? One could place an upper bound on the total hint rate, and keep dropping less important hints if needed or queuing them for delivery at a later point in time (provided they are still valid). Ideally, the user himself should be able to adjust the intrusiveness of a stream of hints.

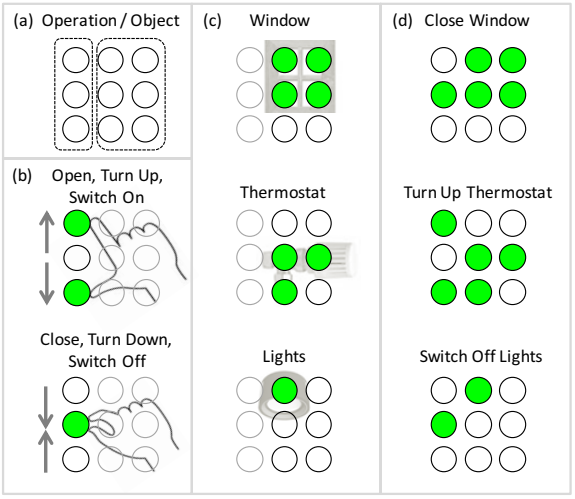

Figure 5. Hint pattern structure using a 3x3 LED display: (a) LED allocation; (b) operations; (c) target objects; (d) example hints.

D. *Control-theoretic issues*

Unreliable actuation. A sense-and-react feedback loop is a control system, with the context-aware application acting as the controller. The loop with soft actuation gives rise to potentially interesting control-theoretic problem: to work out *control laws* that take into account both (a) the uncertainty of actuation (given that it is up to the user) and (b) the lack of direct information on whether the actuation has occurred (the controller may try to infer that by sensing the environment).

VI. CONCLUSION

To validate soft actuation, we are setting up an experiment within the European FP7 SmartSantander project [20]. The experiment will use the IoT testbed deployed at the University of Surrey (UK). It will involve 15 participants exposed to a soft actuating system during their everyday activities for about five weeks. A context-aware application will use sensors placed in participants' offices to measure temperature, brightness, noise, and motion. Based on this data, the system will issue six hints from Table II (based on rationales R1-R11). The users will be providing feedback on their reaction to individual hints; at the end, they will complete a questionnaire and take an interview.

The hints will be delivered via the earlier-mentioned panel featuring 3x3 grid of LEDs, without a buzzer. The hint pattern design and some examples are shown in Fig. 5. The left column of the LED grid is used to encode the operation, while the remaining two columns – the target object (see Fig. 5a). For both encodings we use LED combinations of some mnemonic value. The generic "open" and "close" operations are encoded in the spirit of the hand gestures for "stretch" and "shrink," known from touch screens (see Fig. 5b). Similarly, the object encodings bear some (even if quite remote) resemblance to the respective physical objects (see Fig. 5c).

The experiment, as well as future work, will hopefully resolve a number of research challenges identified above. We are highly motivated to pursue this path, as soft actuation, in spite of (or because of) its apparent simplicity and low profile, seems to be a very promising way to actually introduce pervasive computing into the home and office. The technique appears widely applicable, in terms of the variety of possible hints, the diversity of possible context-aware infrastructures, as well as the ease and low cost of simple deployments.

ACKNOWLEDGMENT

This work was funded in part by the European Commission, project SmartSantander, contract nr. FP7-ICT-257992. We thank Aleksander Pruszkowski for proposing the grid arrangement of LEDs in the display.

REFERENCES

[1] Bellotti, V., Edwards, W. K.: Intelligibility and Accountability: Human Considerations in Context-Aware Systems, in Human-Computer Interaction, 16, 2, pp. 193-212, 2001.

[2] Intille, S. S.: Designing a Home of the Future, in IEEE Pervasive Computing, 1, (2), pp. 76-82, 2002.

[3] Streitz, N. A., Rocker, C., Prante, Th., van Alphen, D., Stenzel, R., Magerkurth, C.: Designing Smart Artifacts for Smart Environments, in IEEE Computer 38, 3, pp. 41-49, 2005.

[4] Makonin, S., Bartram, L., Popowich, F.: A Smarter Smart Home: Case Studies of Ambient Intelligence, in IEEE Pervasive Computing, 12, 1, pp. 58 – 66, 2013.

[5] Weiser, M., Brown, J. S.: Designing Calm Technology, in PowerGrid Journal, vol. 1, 1996.

[6] Danfoss Heating Solutions: living eco® Installation and User Guide, http://heating.consumers.danfoss.com

[7] Ackerly K., Brager G.: Window Signaling Systems: Control Strategies & Occupant Behavior, Proc. 7th Windsor Conference: The changing context of comfort in an unpredictable world, 2012.

[8] Seligman, C., Darley, J. M., Becker, L. J.: Behavioral approaches to residential energy conservation, in Energy and Buildings, 1, 3, April 1978, pp. 325-337, 1978.

[9] Calemis, I., Goumopoulos, G., Kamneas, A.: Talking Plant: Integrating Plants Behavior with Ambient Intelligence, Intl Conf. on Intelligent Environments, pp. 335-343, 2006.

[10] Chen, G., Kotz, D.: A survey of context-aware mobile computing research, TR2000-381, Department of Computer Science, Darthmouth College, 2000.

[11] Pousman, Z., Stasko, J.: A Taxonomy of Ambient Information Systems: Four Patterns of Design, ACM Working Conf. on Advanced Visual Interfaces, pp. 67-74, 2006.

[12] Kaushik, P., Intille, S. S., Larson, K.: User-adaptive Reminders for Home-based Medical Tasks – A Case Study, in Methods of Information in Medicine, 47, (3), pp. 203-207, 2008.

[13] Fogg, B. J.: Persuasive Computers: Perspectives and Research Directions, ACM CHI, pp. 225-232, 1998.

[14] Ratti, C., Nabian, N.: The City to Come, in Innovation: Perspectives for the 21st Century, BBVA, 2010

[15] McFarlane, D. C., Latorella, K. A.: The scope and importance of human interruption in human-computer interaction design, in Human-Computer Interaction, 17, 1, pp. 1-61, 2002.

[16] Quinn, A. J., Bederson, B. B.: Human Computation: A Survey and Taxonomy of a Growing Field, ACM CHI, pp. 1403-1412, 2011.

[17] AmbientOrb, http://www.ambientdevices.com/

[18] Blattner M. M., Sumikawa D. A., Greenberg R. M.: Earcons and icons: their structure and common design principles, in Human-Computer Interaction, 4, 1, pp. 11-44, 1989.

[19] Matthews T., Dey A. K., Mankoff J., Carter S., Rattenbury T.: A toolkit for managing user attention in peripheral displays, Proc. 17th ACM Symp. on User Interface Software and Technology, pp. 247-256, 2004.

[20] SmartSantander Project Page, http://www.smartsantander.eu/