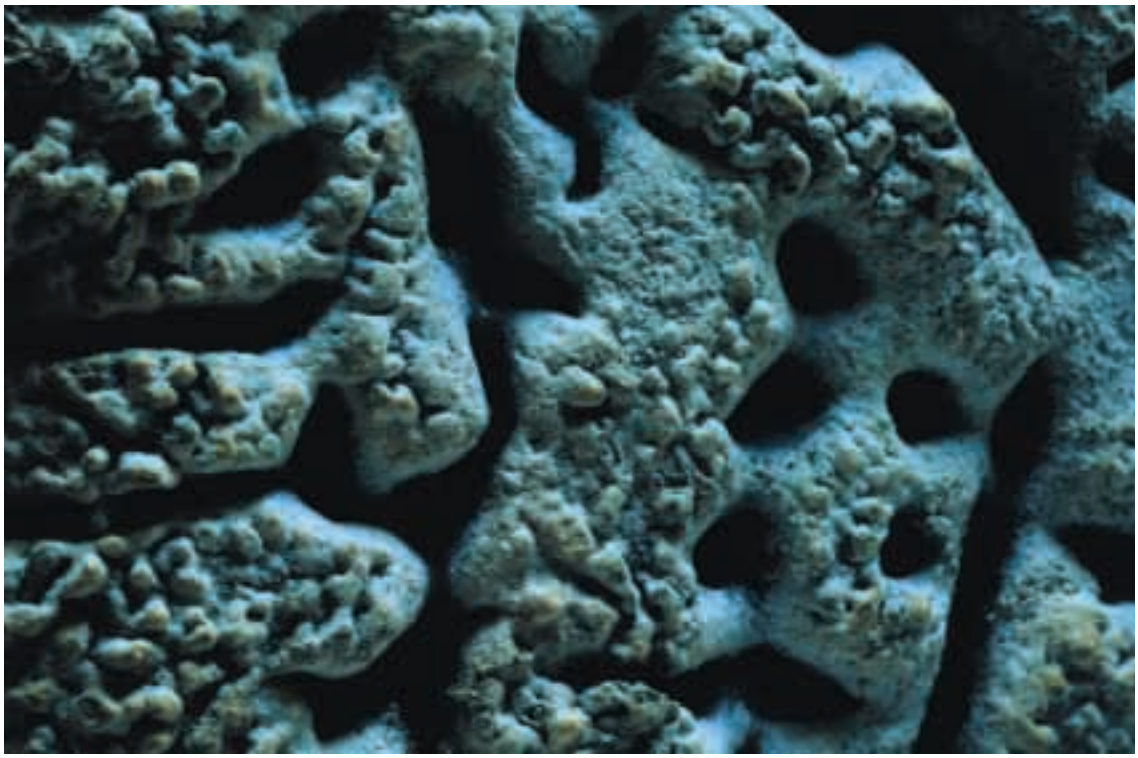


David Wasserman ©1997 Artville, LLC

The Activity Checklist: A Tool for Representing the "Space" of Context

Introduction

In recent years, specialists in human–computer interaction (HCI) have come to appreciate the importance of understanding the context in which computer-supported activities take place [1]. Such understanding directly affects design and evaluation by revealing what users are up to and how they might most effectively use a technology. The idea is to gain this understanding before the design process has progressed too far, or during evaluation, when openings for modifications and improvements to the technology exist .

Victor Kaptelinin
Department of Informatics
Umeå University
S-901 87 Umeå, Sweden
Victor.Kaptelinin@informatik.umu.se

Bonnie A. Nardi
Department of Human–Computer Interaction
AT&T Labs–Research
Menlo Park, California 94040, USA
nardi@research.att.com

Catriona Macaulay
Napier University
Department of Computing
Edinburgh, EH14 1DJ, United Kingdom
c.macaulay@dcs.napier.ac.uk

There have been several attempts to come up with tools and techniques to support taking context into account in the design and evaluation of computer technologies. These approaches include task analysis [6], participatory design [3], and contextual design [7], among others. However, contextual factors are notoriously elusive and difficult to pin down [5], so there is still a need for conceptual tools to deal with context at a practical level.

The existing approaches to context are for the most part "bottom up" ones. They start with an empirical analysis of contextual factors and gradually develop concepts such as "task decomposition" [6], "future workshops" [3], or "flow models" [7], which later can be put in an appropriate theoretical framework. From our point of view, this "bottom up" or empirically-driven strategy can be complemented with a "top down" one, that is, starting with an abstract theoretical representation of context and then situating this representation in the reality of design and evaluation. Borrowing Brown and Duguid's well-known metaphor [5], we can say that if it is difficult to grapple with the "whale" of context by trying to get a firm grip on its specific parts, let's try a large net instead.

In this paper we present a tool that is directly shaped by a general theoretical approach—activity theory [10, 11, 18]. Activity theory provides a broad theoretical framework for describing the structure, development, and context of human activity. In the 1990s, activity theory has been applied to problems of human–computer interaction by an international community of scholars and practitioners [1–4, 8, 9, 12].

Activity theory is framed by several basic principles (explained in the next section): hierarchical structure of activity, object-orientedness, internalization and–externalization, tool mediation, and development. These general principles help orient thought and research, but they are somewhat abstract when it comes to the actual business of working on a design or performing an evaluation. To make activity theory more useful, we have developed an artifact—the Activity Checklist—that makes concrete the conceptual system of activity theory for the specific tasks of design and evaluation.

The Activity Checklist is intended to elucidate the most important contextual factors of human–computer interaction. It is a guide to the specific areas to which a researcher or practitioner should be paying attention when trying to understand the context in which a tool will be or is used. The Checklist lays out a kind of "contextual design space" by representing the key areas of context specified by activity theory.

In the rest of this paper we discuss activity theory, present the Checklist, and show its use by giving an example of a specific technology. The Checklist is an adjunct to the basic principles of activity theory—not a tool to be used in isolation. An overview of activity theory with empirical applications can be found in [13].

Basic Principles of Activity Theory: An Overview

Activity theory is a general conceptual approach, rather than a highly predictive theory. The unit of analysis in activity theory is the activity, consisting of a subject (an individual or group), an object or motive, artifacts, and sociocultural rules. Leont'ev [10] made the point that we cannot pull these pieces apart without violating the very essence of human activity, just as we cannot pull apart sodium and chloride if we want to understand salt. Understanding human activity requires a commitment to a complex unit of analysis.

Two basic ideas animate activity theory: (1) the human mind emerges, exists, and can only be understood within the context of human interaction with the world; and (2) this interaction, that is, *activity*, is socially and culturally determined. These ideas are elaborated in activity theory into a set of five principles as follows.

Object-Orientedness

The principle of object-orientedness states that every activity is directed toward something that objectively exists in the world, that is, an *object*. For example, a computer program is an object of a programmer's activity.

Human activity can be oriented toward two types of objects: things and people [10]. The notion of an object is not limited in activity theory to the physical, chemical, and biological properties of entities. Socially and culturally determined properties are also objective properties that can be studied with objective methods. For example, the intended purposes and ways of using artifacts can be objectively studied.

Hierarchical Structure of Activity

According to Leont'ev [11], interaction between human beings and the world is organized into functionally subordinated hierarchical levels. Leont'ev differentiated among three levels: activities, actions, and operations. Activities are undertaken in order to fulfill *motives.* Motives can be considered top-level objectives that are not subordinated to any other objectives. Behind a motive "… there always stands a need or a desire, to which [the activity] always answers" [10]. People may or may not be consciously aware of their motives.

Actions are goal-directed processes that must be carried out to fulfill a motive. For instance, a programmer may write a utility program needed to make his larger program work efficiently. The larger program itself might be an action with respect to a motive such as getting ahead at work. Actions are conscious; people are aware of their goals.

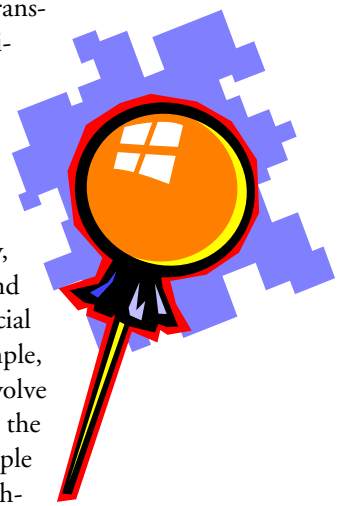
Goals can be broken into lower level goals, which, in turn, can have lower level goals, much like the concept of goals and subgoals in artificial intelligence (AI) and other traditions. For example, writing a utility program might involve talking to another programmer about how she solved a similar problem, which might involve scheduling a time to talk, opening an electronic calendar, and so forth. Actions are similar to what are often referred to in the human–computer interaction literature as *tasks* [e.g., 15].

Moving down the hierarchy of actions we cross the border between conscious and automatic processes. Functional subunits of actions, which are carried out automatically, are *operations.* Operations do not have their own goals; rather they adjust actions to current situations. Actions transform into operations when they become routinized and unconscious with practice. When learning to drive a car, the shifting of the gears is an *action* with an explicit goal that must be consciously attended to. Later, shifting gears becomes *operational* and "can no longer be picked out as a special goal-directed process: its goal is not picked out and discerned by the driver" [10]. Conversely, an operation can become an action when "conditions impede an action's execution through previously formed operations" [10]. For example, if one's mail program ceases to work, one continues to send mail by substituting another mailer, but it is now necessary to pay conscious attention to using an unfamiliar set of commands. This dynamic movement up and down the hierarchy distinguishes the activity theory hierarchy from static models such as GOMS.

Internalization and Externalization

Activity theory differentiates between internal and external activities. The traditional notion of mental processes (such as in cognitive science) corresponds to internal activities. Activity theory emphasizes that internal activities cannot be understood if they are analyzed separately, in isolation from external activities, because it is the constant transformation between external and internal that is the very basis of human cognition and activity.

Internalization is the transformation of external activities into internal ones. Activity theory emphasizes that not only do mental representations get placed in someone's head, but the holistic activity, including motor activity and the use of artifacts, is crucial for internalization. For example, learning to calculate may involve counting on the fingers in the early stages of learning simple arithmetic. Once the arith-



Methods & Tools
Column Editors

Michael Muller
Lotus Development Corp.
55 Cambridge Parkway
Cambridge, MA 02142 USA
+1-617-693-4235
fax: +1-617-693-1407
mullerm@acm.org

Finn Kensing
Computer Science
Roskilde University
P.O. Box 260
DK-4000 Roskilde
Denmark
+45-4675-7781-2548
fax: +45-4674-3072
kensing@dat.ruc.dk

metic is internalized, the calculations can be performed in the head without external aids. Internalization provides a means for people to try potential interactions with reality without performing actual manipulation with real objects (mental simulations, imaginings, considering alternative plans, and so forth). Therefore, internalization can help identify optimal actions before actually performing an action externally. In some cases, internalization can make an action more efficient because external components, such as performing calculations in the head, are omitted.

Externalization transforms internal activities into external ones. Externalization is often necessary when an internalized action needs to be repaired, or scaled, such as when a calculation is not coming out right when done mentally or is too large to perform without pencil and paper or calculator (or some external arti-

The Checklist in the Field

Catriona Macaulay

As a tool for thinking, the Checklist lends itself to many situations and uses. In this section I illustrate one such situation—the domain investigation—with a personal account of my experiences using the Checklist.

For some time now, the need to "contextualize" the design of computer systems has been recognized [1]. Context is of course a notoriously slippery term, and contextualizing design can mean anything from simply taking into account the physical environment in which a system is to be used to developing richly detailed accounts of how people do the things we design new artifacts to support. Ethnographic techniques (see [3] for an introduction) have become firmly established as one way of gathering contextual information. The uses of ethnography within design settings have been described as a continuum, ranging from requirements gathering tied to a particular development project, to opening up a broad domain such as information gathering in order to contribute to our currently limited understanding about fundamental tasks. [4]

My field site was a UK national daily newspaper. I had gone there to explore what 'information gathering' meant in the context of journalism. And I was doing this for a very explicit purpose, that of informing the design of future technologies to support such activities. Like many ethnographers, having made the decision to go into the field I was unsure about what to do when I got there. To complicate matters, I came from a background in computing and human-computer interaction studies and therefore was taking a particular information technology-biased set of preconceptions and inclinations into the field with me.

These issues, my natural inclinations towards theory, and my inexperience as a fieldworker all led me to look for some kind of theoretical scaffolding. Activity Theory (AT) seemed a good choice. AT, I reasoned, had been investigated within HCI and CSCW circles for some time. [2] It seemed to offer hope for bridging the field-design gap by providing a set of concepts relevant to both AT researchers and designers. And activity theory provided a particularly rich set of insights into the relationship between artifacts and practice.

The adoption of theoretical frameworks is, of course, not without its dangers. Prior to conducting my main study, I undertook a short pilot study at a small community organization. Very quickly during this period I felt overwhelmed by my attempts to orient my field experiences around activity theory issues, and I eventually abandoned the attempt. It was at this point that I fortuitously discovered the Checklist. Now I had something *tangible* I could use. It gave me a quick way of relating experiences in the field to AT concepts. It helped me think about the kinds of data I wanted to gather, and the kinds of questions I wanted to ask. As time went on, field driven concerns came to dominate my efforts and the Checklist took more of a back seat. When I was out of the field and reviewing my notes and transcripts, the Checklist once again gave me an additional viewpoint on it all.

But how did I actually use it? Well, one of the Checklist's benefits is its informality. Key concepts are illustrated with sample questions which suggest avenues for thought and exploration rather than formal directions. The Checklist orients without prescribing. I reduced the main section of the Checklist to A5 and kept a copy in my fieldnotes books as an aide memoir. This proved particularly handy for the nervous neophyte fieldworker I was. It gave me something to look at and think about in those awful moments sitting around in the field feeling completely lost! I also had a copy stuck on my office wall which I could refer to when I was preparing for interviews or observation sessions. During data analysis, the Checklist provided a

fact). Externalization is also important when collaboration between several people requires their activities to be performed externally in order to be coordinated.

Mediation

Activity theory's emphasis on social factors and on the interaction between people and their environments explains why the principle of tool *mediation* plays a central role. First, tools shape the way human beings interact with reality. Shaping external activities results in shaping internal ones. Second, tools usually reflect the experience of other people who tried to solve similar problems before and invented or modified the tool to make it more efficient and useful. This experience is accumulated in (1) the structural properties of tools (shape, size, material) and (2) the knowledge of how the tool should be used. Point 2

resource for deriving additional codewords for my data analysis work.

Just having a copy visible when I was working that I could occasionally look up at was helpful. The sample questions were particularly useful. For example, one day I caught sight of one of the sample questions under the Environment column:

Are concepts and vocabulary of the system consistent with the concepts and vocabulary of the domain?

I suddenly realized that whilst in computing people talk about "information" all the time, in journalism people talk about "sources." Computing people design systems primarily to help people find information, but "information" is often treated by designers simplistically. Journalists, on the other hand, are more interested in the sources of information than in the information itself. The challenge is finding a source for information about something within an extremely limited timeframe. Subjective judgements about the relevance of a piece of information, then, are made largely in relation to judgements about the source. This led me to the realization that sources can be seen as a very particular kind of artifact within journalistic information gathering, and that they have largely been overlooked by designers of information gathering systems.

During the early stages of my study, the sample questions helped me understand the specific issues the Checklist deals with. Later, as my understanding grew, I turned more to the issues in the Checklist rather than the sample questions. Later still, I found myself developing my own sample questions, questions I now carry with me into my next study.

Activity theory and the Checklist also proved a useful counter to my natural inclination to cling to the familiar—to obvious technological artifacts. Entering the world of ethnographic fieldwork from a computing background, one can easily become over-focussed on high-tech devices, or on "information" in a simplistic sense. During my first forays into the field, I was so focussed on what I thought the obvious constituents of information gathering activity would be, that I completely failed to recognize the importance of sources. It was this kind of benefit from the Checklist that I most valued. The Checklist was a *tool for reflexivity*, helping me in my attempts to maintain an awareness of where my own instinctive concerns and interests were closing me off from those of the people I was studying.

In summary then, the Checklist became a valuable aide memoir and a tool for reflexivity. Although the Checklist as presented here does not explicitly draw attention to its reflexive uses, this was clearly something of particular benefit to more broadly scoped fieldwork such as mine. For the theoretically-oriented fieldworker, the Checklist provides a flexible and non-prescriptive way of maintaining an awareness of potentially relevant aspects of AT to design concerns. Of course it does not do away with the need to engage with the ideas behind activity theory more broadly, but it certainly helps kick-start the process.

References

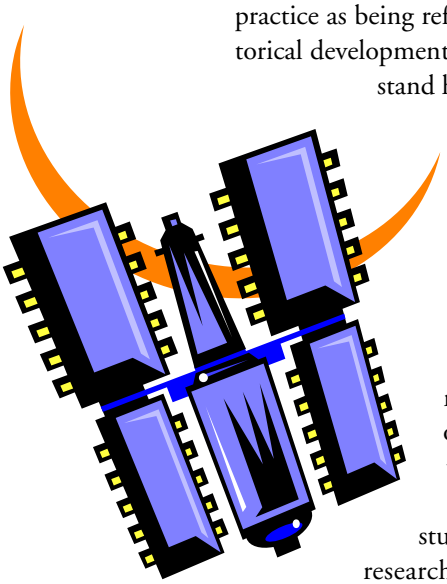
1. Clarke, S. (1997). Encouraging the Effective Use of Contextual Information in Design. Unpublished PhD, University of Glasgow, Glasgow, Scotland.
2. Draper, S. (1992). Activity Theory: The New Direction for HCI? *International Journal for Man-Machine Studies*, 37.
3. Hammersley, M., & Atkinson, P. (1995). *Ethnography: Principles in Practice*. (2nd ed.). London: Routledge.
4. Whittaker, S., Terveen, L. and Nardi, B. Let's stop pushing the envelope and start addressing it. Submitted to *TOCHI*.

is critical for activity theory. Many theories discuss Point 1 (such as the idea of affordances, Latourian notions of tool prescriptions, and so forth). Activity theory emphasizes that a tool comes fully into being when it is *used* and that knowing how to use it is a crucial part of the tool. So, the use of tools is an evolutionary accumulation and transmission of social knowledge, which influences the nature of not only external behavior but also the mental functioning of individuals.

The concept of tool in activity theory is broad and embraces both technical tools, which are intended to manipulate physical objects (e.g., a hammer), and psychological tools, which are used by human beings to influence other people or themselves (e.g., the multiplication table or a calendar).

Development

Finally, activity theory requires that human interaction with reality be analyzed in the context of development. Activity theory sees all practice as being reformed and shaped by historical development. It is important to understand how tools are used not in a single instant of trying them out in a laboratory (for example) but as usage unfolds over time. In that time, development may occur making the tool more useful and efficient than might be seen in a single observation. In activity theory, development is thus not only an object of study, it is also a general research methodology. That is why a basic research method in activity theory is the formative experiment which combines active participation with monitoring of the developmental changes in the object of study.



Integration of the Principles

These basic principles of activity theory should be considered an integrated system, because they are associated with various aspects of the whole activity. A systematic application of any of these principles makes it eventually necessary to involve all the others. For instance, understanding the *hierarchical structure* of an activity requires an analysis of its *object or motive*, as well as *developmental* transformations between actions and operations and between *internal and external* components. The latter, in turn, can critically depend on the *tools* used in the activity.

Activity Checklist

As mentioned earlier, activity theory does not provide ready-made solutions that can be directly applied to specific problems. We see its main potential in supporting researchers and designers in their own search for solutions, in particular, by helping them to ask meaningful questions. To make such an application of activity theory more practical, we introduce an analytical tool, the Activity Checklist.

The Activity Checklist is intended to be used at early phases of system design or for evaluating existing systems. Accordingly, there are two slightly different versions of the Checklist, the "evaluation version" and the "design version." Both versions are used as organized sets of items covering the contextual factors that can potentially influence the use of a computer technology in real-life settings. It is assumed that the Checklist can help to identify the most important issues, for instance, potential trouble spots, that designers can address.

Having two versions of the Checklist implies a commitment to the study of actual use as a critical part of design. Researchers such as Bannon [1] have made the useful point that design and use are two sides of the same coin. Still, a design must begin somewhere, and it is helpful to have guidance in the earliest stages of brainstorming and creative imagining of how a technology might come into being.

The Checklist covers a large space. It is intended to be used first by examining the whole space for areas of interest, then focusing on the identified areas of interest in as much depth as possible. The general strategy, then,

is breadth-first consideration of the relevant areas of context enumerated in the Checklist, followed by a "drilling down" into specific areas that should yield rich results given the tools and problems at hand.

The structure of the Checklist reflects the five basic principles of activity theory. Since the Checklist is intended to be applied in analyzing how people use (or will use) a computer technology, the principle of *tool mediation* is strongly emphasized. This principle has been applied throughout the Checklist and systematically combined with the other four principles. It results in four sections corresponding to four main perspectives on the use of the "target technology" to be evaluated or designed:

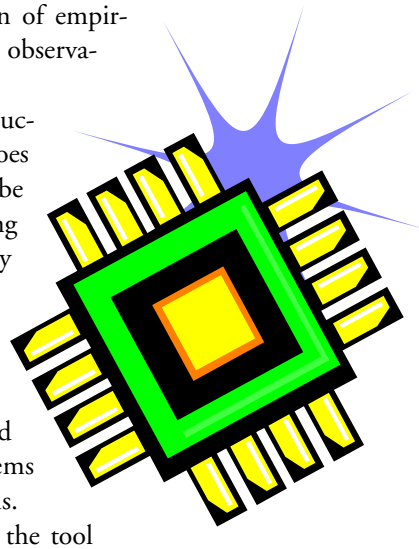
1. **Means and ends**—the extent to which the technology facilitates and constrains the attainment of users' goals and the impact of the technology on provoking or resolving conflicts between different goals.
2. **Social and physical aspects of the environment**—integration of target technology with requirements, tools, resources, and social rules of the environment.
3. **Learning, cognition, and articulation** — internal versus external components of activity and support of their mutual transformations with target technology.
4. **Development**—developmental transformation of the foregoing components as a whole.

Taken together, these sections cover various aspects of how the target technology supports, or is intended to support, human actions ("target actions"). See the Checklist in the Appendix to this paper.

Using the Checklist

According to our experience of using the Checklist and teaching other people how to use it, there are several points to remember when trying to apply the tool in a specific project.

First, the Checklist is supposed to be used not as the only basis for system design or evaluation, but in combination with other techniques. One of the main advantages of using the Checklist seems to be more effective application of a number of already established methods and techniques. For instance, the Checklist can help identify the most relevant issues to be covered in an interview or to make sure important problems are not overlooked in a discussion of empirical data collected in an observational study.



Second, the linear structure of the Checklist does not imply that it should be used linearly, by focusing on isolated items one by one and ignoring the rest of the Checklist. Instead, practitioners using the tool should look for patterns of related items, even if these items belong to different sections.

Third, in order to use the tool effectively, practitioners should familiarize themselves with the Checklist and even try to internalize it. We recommend that practitioners follow the items in the Checklist repeatedly at various phases of design or evaluation. A quick initial run should identify the most important potential trouble spots and filter out the rest. Further runs may result in finding patterns, revising previously made judgments about the importance or unimportance of certain issues, and formulating requests for more information, if necessary.

Fourth, it should be noted that every tool is used for some purpose, and the Checklist is no exception. Therefore, potential users of the Checklist should clearly understand why they are using the tool. Such understanding can help focus on relevant items and ignore irrelevant ones. Also, such understanding is necessary for successful incorporation of conclusions, judgments, and ideas related to individual items into more general notions relevant to design or evaluation of the system as a whole.

Apple Data Detectors: An Example of Using the Checklist

The design of Apple Data Detectors, a multi-

purpose intelligent agent technology for analyzing and taking action on structured data [14], is an example of using the Checklist to go beyond the narrow scope of many design projects. Apple Data Detectors recognizes structured data such as URLs, e-mail addresses, postal addresses, ISBN numbers, and stock symbols. Using "structure detection" technology [17], a detector is written to analyze structured data. The detector is then paired with an action such as "Open URL" or "Create e-mail message." Apple Data Detectors works in any text; applications do not need to be modified to use it.

In the design of Apple Data Detectors, we were concerned with the Learning/development areas of the Checklist. We devoted many resources to considering how end users would go from simple use of the tool involving only accessing structures supplied by Apple or third-party developers to programming their own new agents. As we considered the principle of *development* from the beginning of our research, we were able to create an architecture that supports end-user programming [12]. We paid special attention to the first four areas in the Development column. To reiterate the point made earlier, the Checklist can be used to scope out a large possible space of potential areas of interest and then narrow down to specific areas to actively pursue. The Checklist is useful in reminding developers of a larger space that gets beyond the details of user interface mechanisms and leads systematically into many areas of the context of use that may provide inspiration for interesting designs.

A research project at another lab used structure detection technology much as we did [16]. But the prototype looked quite different because the emphasis was not the users' wider context as it was in Apple Data Detectors. Apple Data Detectors allows for end-user development of structures, scripting of actions, mixing and matching recognizers and actions, and composite structures (see Table 1). It provides flexibility and a growth path for users.

Most designers will have to be concerned with the Means/ends column of the Checklist. In Apple Data Detectors we studied potential uses of Data Detectors and found that for the technology to be useful, users need composite structures such as postal addresses. It is considerably more difficult to write a parser that handles composite structures (e.g., an address is composed of a name, street, city, etc., each of which is an atomic structure). But our users can select an address (with the mouse), and Apple Data Detectors will recognize it and take a prespecified action such as adding the address to the user's address book, putting each field of the address in the appropriate place in the address book.

We also gave careful thought at the outset of the project to our criteria for success and failure (the Design section of the Means/ends column). Our criteria were that the technology be useful for Apple customers and that developers be able to use it painlessly. (Third-party developers are developing the structures and actions that work with their applications.) We thus decided not to use OpenDoc, even though there was some pressure to do so. But the developers' experience would have been much more difficult. As it turned out, this was the right decision more than we knew at the time, because OpenDoc was eventually put on the corporate back burner. It was the explicit

Table 1 A COMPARISON OF TWO USES OF STRUCTURE DETECTION TECHNOLOGY

Intel Selection Recognition Agent	Apple Data Detectors
No scripting (C behind an API)	AppleScript
No path to end-user modification	Script editor End-user modification
Recognizer/action pairs bound together	Separate recognizers, actions
No composite structures	Composite structures

attention to a firm set of design criteria that helped us weather that storm.

Conclusion

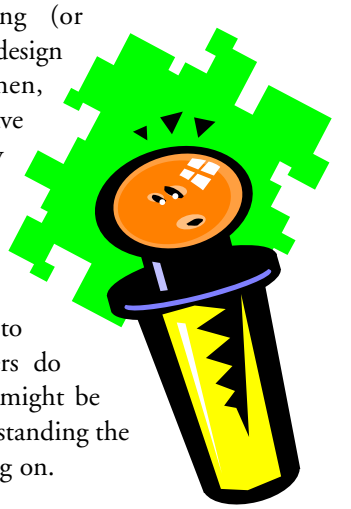
As mentioned earlier, the Activity Checklist is not the only attempt to deal with context in the field of HCI, and it is not intended as a substitute for other approaches. From our point of view, the Checklist can be most successfully used together with other tools and techniques to efficiently address issues of context.

For instance, task analysis [6] places a heavy emphasis on the Means/ends dimension of context, whereas environment and, especially, learning and development are underrepresented. Using the terminology of activity theory, we could say that task analysis gives a thorough description of individual actions, whereas the higher levels of activity and interrelations between actions receive less attention.

Contextual design [7], conversely, provides an elaborated set of concepts and techniques for describing the environment (as in the "Environment" section of the Checklist). It also supports identifying users' tasks (although, in our opinion, not to the extent to which it addresses the structure and functioning of the environment) but is less focused on learning and development. Finally, Development is a major concern within participatory design approaches [e.g., 3], but identifying task structures does not have a high priority there.

Therefore, each of the existing empirically driven approaches to context has its strong points. From our point of view, the main advantage of the Activity Checklist is that it is a general framework that can be used to (1) provide a preliminary overview of potentially relevant contextual factors, (2) select appropriate tools for further exploration, and (3) evaluate limitations of those tools. In other words, the Checklist can help to leverage the various strengths of empirically based approaches.

The fact that the Checklist is comprehensive and wide-ranging should not mislead its potential users. It would be impossible to investigate all the areas it covers without a multiyear study, but that is not how it is intended to be employed. For most uses of the Checklist, users should first do a "quick-and-dirty" perusal of the areas represented in the Checklist that are likely to be troublesome or interesting (or both) in a specific design or evaluation. Then, once those areas have been identified, they can be explored more deeply. The breadth of coverage in the Checklist will help to ensure that designers do not miss areas that might be important for understanding the tool they are working on.



Acknowledgments

Many thanks to Helen Hasan, Mark Spasser, Clay Spinuzzi, and John Waterworth for their helpful comments on an earlier version of the paper.

References

1. Bannon, L. From human factors to human actors: The role of psychology and human–computer interaction studies in system design. In *Design at Work: Cooperative Design of Computer Systems,* J. Greenbaum and M. Kyng, eds., Lawrence Erlbaum, Hillsdale, NJ, 1991.

2. Bødker, S. *Through the Interface: A Human Activity Approach to User Interface Design.* Lawrence Erlbaum, Hillsdale, NJ, 1991.

3. Bødker, S., Knudsen, J., Kyng, M., Ehn, P., and Madsen, K.H. Computer Support For Cooperative Design. In *Proceedings of ACM CSCW'88 Conference on Computer-Supported Cooperative Work* (Portland, OR, 1988).

4. Bourke, I., Verenikina I., and Gould, E. Interacting With Proprietary Software Users: An Application for Activity Theory? In *Proceedings East-West International Conference on Human-Computer Interaction* (Moscow, August 3–7, 1993,). ICSTI, Moscow.

5. Brown, J., and Duguid, P. Borderline issues: Social and material aspects of design. *Human-Computer Interaction 9,* 1 (1994).

6. Dix, A., Finlay, J. Abowd, G., and Beale, R. *Human-*

Computer Interaction. Prentice Hall, London, 1993.
7. Holtzblatt, K., and Beyer, H. Making customer-centered design work for teams. *Communications of the ACM 36,* 10 (1993).
8. Kaptelinin, V. Human–Computer Interaction in Context: The Activity Theory Perspective. In *Proceedings of the East-West Human–Computer Interaction Conference* (St. Petersburg, Russia, August 4–8). ICSTI, Moscow, 1992.
9. Kuutti, K. Activity theory and its applications in information systems research and design. In *Information Systems Research Arena of the 90's*, H.-E. Nissen, H.K. Klein, and R. Hirschheim, eds., Elsevier, Amsterdam, 1991.
10. Leont'ev, A.N. *Activity, Consciousness, Personality.* Prentice Hall, Englewood Cliffs, NJ, 1978.
11. Leont'ev, A. N. *Problems of the Development of Mind.* Progress, Moscow, 1981.
12. Lieberman, H., Nardi, B. and Wright, D. Training Agents to Recognize Text by Example. In *Proceedings of the International Conference on Autonomous Agents,* Seattle, April 1999.
13. Nardi, B., ed. *Context and Consciousness: Activity Theory and Human–Computer Interaction.* MIT Press, Cambridge, MA, 1996.
14. Nardi, B., Miller, J., and Wright, D. Collaborative, Programmable Intelligent Agents. In *Communications of the ACM* March 1998.
15. Norman, D. Cognitive Artifacts. In Designing *Interaction: Psychology at the Human-Computer Interface,* J. Carroll. ed., Cambridge University Press, Cambridge, 1991.
16. Pandit, M., and Kalbag, S. The Selection Recognition Agent: Instant Access to Relevant Information and Operations. In *Proceedings of Intelligent User Interfaces '97.* (1997 Orland, Fl.), New York, ACM Press.
17. Rus, D. and Subramanian, D. Multi-media RISSC Informatics. In *Proceedings of the 2nd International Conference on Information and Knowledge Management.* (Washington, DC, 1993.). ACM Press, New York, pp. 283–294.
18. Wertsch, J. Ed., *The Concept of Activity in Soviet Psychology.* M. E. Sharpe, Armonk, NY, 1981.

Appendix. Activity Checklist

PREAMBLE				
	Means/ends (hierarchical structure of activity)	**Environment** (object-orientedness)	**Learning/cognition/ articulation** (externalization/ internalization)	**Development** (development)
	Human beings have hierarchies of goals that emerge from attempts to meet their needs under current circumstances. Understanding the use of any technology should start with identifying the goals of target actions, which are relatively explicit, and then extending the scope of analysis both "up" (to higher-level actions and activities) and "down" (to lower level actions and operations).	Human beings live in the social, cultural world. They achieve their motives and goals by active transformation of objects in their environments. This section of the checklist identifies the objects involved in target activities and constitutes the environment of the use of target technology.	Activities include both internal (mental) and external components which can transform into each other. Computer systems should support both internalization of new ways of action and articulation of mental processes, when necessary, to facilitate problem solving and social coordination.	Activities undergo permanent developmental transformations. Analysis of the history of target activities can help to reveal the main factors influencing the development. Analysis of potential changes in the environment can help to anticipate their effect on the structure of target activities.

EVALUATION VERSION				
	Means/ends	**Environment**	**Learning/cognition/ articulation**	**Development**
	People who use the target technology Goals and subgoals of the target actions (target goals) Criteria for success or failure of achieving target goals Decomposition of target goals into subgoals Setting of target goals and subgoals Potential conflicts between target goals Potential conflicts between target goals and goals associated with other technologies and activities Resolution of conflicts between various goals Integration of individual target actions and other actions into higher-level actions Constraints imposed by higher-level goals on the choice and use of target technology Alternative ways to attain target goals through lower-level goals. Troubleshooting strategies and techniques Support of mutual transformations between actions and operations	Role of target technology in producing the outcomes of target actions Tools, other than target technology, available to users Integration of target technology with other tools Access to tools and materials necessary to perform target actions Tools and materials shared between several users Spatial layout and temporal organization of the working environment. Division of labor, including synchronous and asynchronous distribution of work between different locations Rules, norms, and procedures regulating social interactions and coordination related to the use of target technology	Components of target actions that are to be internalized Knowledge about target technology that resides in the environment and the way this knowledge is distributed and accessed Time and effort necessary to master new operations Self-monitoring and reflection through externalization Use of target technology for simulating target actions before their actual implementation Support of problem articulation and help request in case of breakdowns Strategies and procedures of providing help to other users of target technology Coordination of individual and group activities through externalization Use of shared representation to support collaborative work Individual contributions to shared resources of group or organization	Use of target technology at various stages of target action "life cycles"—from goal setting to outcomes Effect of implementation of target technology on the structure of target actions New higher-level goals that became attainable after the technology had been implemented Users' attitudes toward target technology (e.g., resistance) and changes over time Dynamics of potential conflicts between target actions and higher-level goals Anticipated changes in the environment and the level of activity they directly influence (operations, actions, or activities)

DESIGN VERSION				
	Means/ends	**Environment**	**Learning/cognition/ articulation**	**Development**
USE	People who use the target technology Goals and subgoals of the target actions (target goals) Criteria for success or failure of achieving target goals Decomposition of target goals into subgoals Setting of target goals and subgoals Potential conflicts between target goals Potential conflicts between target goals and goals associated with other technologies and activities Resolution of conflicts between various goals Integration of individual target actions and other actions into higher-level actions Constraints imposed by higher-level goals on the choice and use of target technology Alternative ways to attain target goals through lower-level goals. Troubleshooting strategies and techniques Support of mutual transformations between actions and operations Goal that can be changed or modified, and goals that have to remain after new technology is implemented	Role of existing technology in producing the outcomes of target actions Tools, available to users Integration of target technology with other tools Access to tools and materials necessary to perform target actions Tools and materials shared between several users Spatial layout and temporal organization of the working environment. Division of labor, including synchronous and asynchronous distribution of work between different locations Rules, norms, and procedures regulating social interactions and coordination related to target actions	Components of target actions that are to be internalized Time and effort necessary to learn how to use existing technology Self-monitoring and reflection through externalization Possibilities for simulating target actions before their actual implementation. Support of problem articulation and help request in case of breakdowns Strategies and procedures of providing help to collegues and collaborators Coordination of individual and group activities through externalization Use of shared representation to support collaborative work	Use of tools at various stages of target action "life cycles"—from goal setting to outcomes Transformation of existing activities into future activities supported with the system History of implementation of new technologies to support target actions Anticipated changes in the environment and the level of activity they directly influence (operations, actions, or activities) Anticipated changes of target actions after new technology is implemented

DESIGN VERSION

	Means/ends	Environment	Learning/cognition/ articulation	Development
DESIGN	Parties involved in the process of design Goals of designing a new system Criteria of success or failure of design Potential conflicts between goals of design and other goals (e.g., stability of the organization, minimizing expenses)	Resources available to the parties involved in design of the system Rules, norms, and procedures regulating interaction between the parties	Representations of design that support coordination between the parties Mutual learning of the content of the work (designers) and possibilities and limitations of technology (users)	Anticipated changes in the requirements to the system

SAMPLE QUESTIONS

	Means/ends	Environment	Learning/cognition/ articulation	Development
	Are all target actions actually supported? Is there any functionality of the system that is not actually used? If yes, which actions were intended to be supported with this functionality? How do users perform these actions? Are there actions, other than target actions, that are not supported, but users obviously need such support? Are there conflicts between different goals of the user? If yes, what are the current trade-offs and rules or procedures for resolving the conflicts? What are the basic limitations of the current technology? Is it necessary for the user to constantly switch between different actions or activities? If yes, are there "emergency exits" which support painless transition between actions and activities, and, if necessary, returning to previous states, actions, or activities?	Are concepts and vocabulary of the system consistent with the concepts and vocabulary of the domain? Is target technology considered an important part of work activities? Are computer resources necessary to produce a certain outcome integrated with each other? Is target technology integrated with other tools and materials? Are characteristics of target technology consistent with the nature of the environment (e.g., central office work vs. teleworking)?	Is the whole "action lifecycle," from goal setting to the final outcome, taken into account and/or supported? Does the system help to avoid unnecessary learning? Is externally distributed knowledge easily accessible when necessary? Does the system provide representations of user's activities that can help in goal setting and self-evaluation? Does the system provide problem representations in case of breakdowns that can be used to find a solution or formulate a request for help? Are there external representations of the user's activities that can be used by others as clues for coordinating their activities within the framework of group or organization?	What are the consequences of implementing the target technology on target actions? Did expected benefits actually take place? Did users have enough experience with the system at the time of evaluation? Did the system require a large investment of time and effort in learning how to use it? Did the system show increasing or decreasing benefits over the process of its use? Are users' attitudes toward the system becoming more or less positive? Are there negative or positive side-effects associated with the use of the system?