Designing for Appropriation: A Theoretical Account

Pierre Tchounikine

Taylor & Francis
Taylor & Francis Group

Human–Computer Interaction

ISSN: 0737-0024 (Print) 1532-7051 (Online) Journal homepage: http://www.tandfonline.com/loi/hhci20

Designing for Appropriation: A Theoretical Account

Pierre Tchounikine

To cite this article: Pierre Tchounikine (2016): Designing for Appropriation: A Theoretical Account, Human–Computer Interaction, DOI: 10.1080/07370024.2016.1203263

To link to this article: http://dx.doi.org/10.1080/07370024.2016.1203263

HUMAN–COMPUTER INTERACTION, 2016, Volume 00, pp. 1–41
Published with license by Taylor & Francis Group, LLC
ISSN: 0737-0024 print / 1532-7051 online
DOI: 10.1080/07370024.2016.1203263

Taylor & Francis
Taylor & Francis Group

Designing for Appropriation: A Theoretical Account

Pierre Tchounikine
University of Grenoble, France

In this article, we propose a theoretical account of software appropriation as a basis for considering how to design software for appropriation. The analysis we develop is based on a holistic perspective including artifact mediation (how software acts as a mediator of users' activities and how users develop instruments), ecologies of artifacts (taking into consideration that users use multiple artifacts), collective and work practice aspects, and technical and software engineering aspects (different forms of and approaches to software adaptation by users). It leads to an understanding that software appropriation is related to how users consider software as a mediator of their activity and turn it into an instrument for themselves based on the functional values they attribute to it. Building on this analysis, We outline a general perspective on designing for appropriation as empowering users to continue software design in use, review some principles and means, and introduce research questions for future work.

1. INTRODUCTION: DESIGNING FOR APPROPRIATION

In this introductory section we first define the notion of software appropriation and introduce what is meant by "designing for appropriation." We then provide an overview of the content and logical organization of the article.

Pierre Tchounikine (Pierre.Tchounikine@imag.fr, http://membres-liglab.imag.fr/tchounikine) is a computer scientist with an interest in the theoretical and software engineering aspects of designing software to support human activity; he is a Professor at the University of Grenoble (France).

CONTENTS

1.1. Software Appropriation

Software appropriation is the process by which users adopt software and adapt it to make it theirs. This includes more than the technical transformation of software by users. Appropriation refers to how technology is adopted in practice (Dourish, 2003).

Different researchers have developed empirical studies of software appropriation, including Mackay's (1990) and Gantt and Nardi's (1992) pioneering ethnographic studies. These studies show that appropriation is a coadaptive phenomenon. Users are not simply passive recipients. Individuals adapt their behavior in response to the technology and adapt the technology to meet their needs. This fundamental idea underlines most works related to appropriation, including ours.

Appropriation is a key concern for human–computer interaction (HCI). People appropriate artifacts in a way that may differ from designers' intentions (Folcher, 2003; Jung, Stolterman, Ryan, Thompson, & Siegel, 2008; Mackay, 1990). Appropriation may thus impact what users use software for and/or how they use it. In such cases, despite requirements analyses and user-centered design (UCD), the artifact may not act as a good mediator and, consequently, may not be adopted or may cause breakdowns (Bødker & Klokmose, 2011).

1.2. Designing for Appropriation

Designing for appropriation consists of intentionally providing means that are likely to enable users to adapt the system to the effective usages they develop. Considering such an objective does not play down the importance of requirements analysis and UCD. However, it suggests that these efforts should be enhanced by considering how users may develop new usages.

Different empirical guidelines have been proposed. Dix (2007) suggested including elements for which users can add their own meanings, exposing intentions, and supporting rather than controlling. Similar suggestions have been made by researchers examining collaborative work (Pekkola, 2003; Robinson, 1993). Mackay (1990) made four suggestions: (a) provide users with feedback about the effectiveness of their adaptation decisions and the opportunity to reflect upon their actual processes of use, (b) allow users to encode patterns of behavior and informally include information about their current work contexts, (c) provide mechanisms for users to share practices, and (d) provide adaptation dimensions that match the users' tasks. The Cognitive Dimensions of Notations framework (Blackwell et al., 2001) raises the general issue of addressing software viscosity (resistance to change). It suggests guidelines such as providing users with secondary notation means, that is, means that allow them to record things that were not anticipated in the design (e.g., tags). One may also consider approaches that evaluate users' experiences (Law, Roto, Hassenzahl, Vermeeren, & Kort, 2009), and aspects such as utility, usability, stimulation, and value. With respect to technical issues, work related to end-user development (EUD) provides different approaches (see Section 5.3).

Nevertheless, as mentioned by Dix (2007), designing for appropriation seems like an oxymoron ("plan for the unexpected"). The main difficulty is how to help designers

identify which aspects of software users may need or want to adapt, and the means that should be provided. Addressing this difficulty is important for two reasons. First, designing adaptable software presents an additional challenge and cost for designers. Second, and most important, offering users adaptation means may directly compromise usability and simplicity. Making it possible to adapt all the elements that can be technically adapted is useless (Mackay, 1990), and may even be counterproductive.

In this article, we propose to address "designing for appropriation" by considering why and how users appropriate software: We propose a theoretical account of appropriation and then study how it informs designing for appropriation. It is not possible to fully predict how users will appropriate software. However, in keeping with authors such as Bødker and Klokmose (2011), Stolterman (2008), and de Souza (2005), we believe that proposing a theoretical account of phenomena—in this case, software appropriation—may help designers prepare for action through a conceptual framework.

1.3. Content and Organization of the Article

To contribute to the better understanding of software appropriation and help designers consider appropriation issues, this article develops four proposals: a general perspective for software appropriation, a theoretical account, a conceptualization of designing for appropriation, and concrete takeaways for technologists and theorists, namely, implications for design, methodological and technical means, and new areas of study.

The article's logical organization and its summary are as follows. First, in Section 2 we propose a human-centric and, more precisely, an activity-centric and developmental perspective on software appropriation. We propose seeing appropriation as the way users consider software as a mediator of their activity and turn software into instruments for themselves, which is related to the functional values they attribute to software. Designing for appropriation thus requires focusing on instruments rather than artifacts, and understanding how they develop. To address this issue, in Section 3 we propose a theoretical account of software appropriation as the development of instruments, which highlights the roles of psychological and social factors, and the fact that instruments are dynamic entities. A consequence is that, whatever the design, there may be tension between the instruments developed by users and the existing artifacts, which may lead or require users to adapt these artifacts. The proposed perspective and theoretical account thus lead us to consider that designing for appropriation should be thought of as creating the conditions for users to use artifacts in line with their personal and/or collective developed instruments, in other words, as a strategy for offering users the means to resolve tensions between instruments and artifacts themselves. To this end, in Section 4 we propose a conceptualization of designing for appropriation as empowering users to "continue software design in use" and a study of the different individual and collective realties that this general idea may take. In Section 5 we then develop the value of these insights in terms of implications for design and new areas of study. This comprises some general principles, a review

of methodological means that may be used to conduct the required analyses and of technical and software engineering means that may be used for implementation, and research questions that this work raises for new areas of study. Finally, in Section 6, we analyze connections with other design approaches and works.

To keep this article focused and consistent with the logical argumentation, references to other works are introduced as they become relevant rather than in preliminary exposition sections. This is not to claim that other perspectives on software appropriation should be discarded and/or deemed inadequate. In fact, the perspective we develop leads to conclusions and design considerations that correlate with other studies, and it may be used with other perspectives to inform design (see the discussion in Section 6).

2. GENERAL PERPSECTIVE: APPROPRIATION AS RELATED TO HOW USERS ATTRIBUTE FUNCTIONAL VALUES TO SOFTWARE

In this section we introduce the notion of functional value, which is central to our theoretical construction. We illustrate it with an example we use throughout the article, and then examine it from an activity-centric perspective.

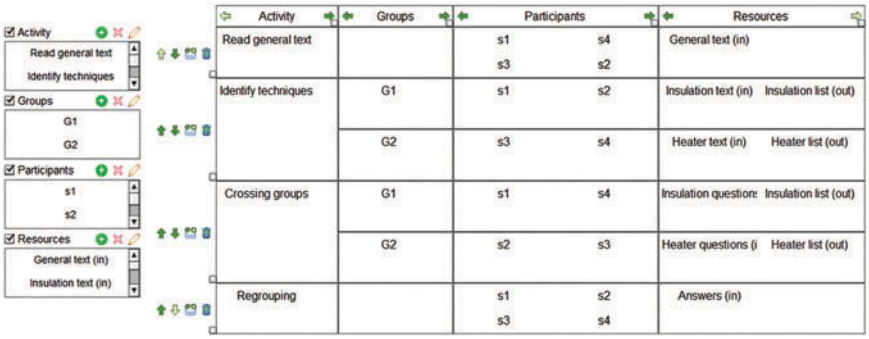
2.1. The Notion of Functional Value

We define the functional value of an artifact for a user (or group of users) as the utility of this artifact for achieving some tasks or goals as perceived by the user (or group of users). Although we use the wording "functional value" as such, it must be kept in mind that a functional value is a value *for* somebody. A piece of software as such does not have a functional value. A user attributes functional values to it when he or she perceives it as a means to achieve tasks that he or she considers.

An important aspect of the notion of functional value is the reference to users' perceptions. Let us clarify this. A piece of software offers technical features. These features can be described in technical terms, their semantics are defined by the system's implementation, and the fact that these features allow a given user or type of user to achieve a given task may be measured by empirical studies. The notion of functional value refers to another perspective on the user–artifact relationship, centered on the user, which is as follows: Users consider certain tasks and, in so doing, consider the means of which they are aware and that they can access. Taking this perspective, what stands out is the user's perception of these means and of their features and convenience. This perception is related, first and foremost, to the tasks the user considers, which may or may not correspond to the purpose for which the software was designed. It is also related to how users address these tasks, which is tied to psychological and/or social constructions (we develop this in Section 3).

We illustrate this idea using a running example of a learning scenario editor, which we introduce next.

FIGURE 1. The scenario editor interface (excerpts from a study; translated into English).



Note. **The editor has an interface similar to that of a table editor in an office suite. It features five conceptual notions as columns (Activities, i.e., students' tasks; Groups; Participants; Resources; and Roles, not used here). Rows may be broken down into subrows using splitting and merging facilities, for example, tasks may be broken down into subtasks attributed to different groups, or different members of a group may receive different resources. Representing a scenario with the editor consists in actions such as defining items (e.g., activities or groups) using the editing boxes on the left, selecting and ordering the notions/columns used (here, the adopted representation structure is Activity–Groups–Participants–Resources), creating rows (with an "Insert a new row" button, not shown here); creating or merging subrows (right click on a cell), putting items in cells (drag-and-drop items from an editing box or another cell). Technically, the table is modeled and implemented as a tree (rows and subrows correspond to the branches of a tree).**

Running Example: The Learning Scenario Editor

The system from which we take examples is a scenario editor (see Figure 1), that is, a system that offers teachers an interface to edit learning scenarios. A learning scenario (scenario, for short) is a model that specifies the tasks (or "learning activities") that a group of students are expected to perform. As an example, the jigsaw scenario pattern is (a) create groups of students (called "expert groups") and provide them with resources (e.g. texts) on different topics, then (b) mix these expert groups in "jigsaw groups" and ask them to accomplish a task requiring information from the different topics. The expectation is that students with different expertise gained in Phase 1 will explain things to each other, debate, and solve conflicts in Phase 2. Figure 1 shows an example edited by a teacher.

Using a pattern such as a jigsaw in a particular setting requires teachers to define students' tasks and subtasks (e.g., if addressing energy saving issues, subtasks could be identifying heating and insulation techniques), roles, groups, given resources, and outputs to be produced. Teachers may implement their pedagogical ideas and deal with local constraints by playing on different registers, for example, task breakdown, group composition, student roles, resources, or data flow. Most of these aspects are interrelated. Consequently, editing a scenario is an elaboration experience that involves defining an initial list of subtasks, splitting a task that seems too complicated, clarifying teacher expectations by defining student roles or group roles, changing the composition of a group to better meet conflicting constraints, and so on.

Throughout the article, we use examples of how teachers considered this editor in relation to the functional values they attributed to it. These illustrative data were taken

FIGURE 2. Examples of Perceived Functional Values for the Scenario Editor.

Teachers and Context		Perceived Functional Values
K-12 teachers; editor presented as a means to represent scenarios using the notions of activity, groups, and so on (see Figure 1, representation of a jigsaw scenario instantiated on insulation/heating topics).	A	having a general plan of what will be done during the session and/or having support to manage groups and activities during the session, that is, a resource for action (this is what the system was designed for)
	B	having a tangible record of what was done in the classroom, as a resource to be annotated and reused for future sessions
	C	sharing pedagogic constructions with colleagues
	D	proving thorough classroom preparation for one's supervisor
Student-teachers in preservice education (part of the year as students and part of the year teaching); editor presented as a means to prepare classroom sessions, using notions from a standard preparation sheet (step, teacher's activity, student's activity, participants, and resources).	E	editing the general structure of classroom activities
	F	(this is what the system was designed for)
	G	checking that student activities planned as parallel sessions would not require the teacher's presence in different groups at the same time
		proposing duration syntheses to help balance the teaching time across subject areas—mathematics, language, and so on—or types of activities

from several in-lab tests and discussions with different teachers in different contexts, including tests with five K-12 teachers, 18 academics, and eight student-teachers; see Sobreira and Tchounikine (2012, 2015) for details. We use this example to illustrate theoretical constructions, not as empirical proof.

Examples of Perception of Unexpected Functional Values

The editor was designed for the following prototypical use case: teachers considering scenarios patterns (e.g., jigsaw) and, while editing an instance (i.e., defining the precise activities or resources), adapting the scenario to their view and teaching context. The tests focused on this use case and ended with an open discussion around how teachers perceived the editor, if and how they would use it, how it resonated with their work practices, and their suggestions and requests.

In Figure 2, we report some of the functional values that teachers perceived as they emerged from these tests. Clearly, teachers perceived what the editor was designed for, but also saw other functional values, some of which are surprising (the table does not report perceptions related to an obviously technically incorrect understanding of the system's features).

The K-12 teachers were all experienced teachers. All of them acknowledged the intended value of the editor (Figure 2, A), although with variations. All of them also developed other perceptions in relation to their work practices and, in particular, improving practices (e.g., B or C). These different functional values are not contradictory and may correspond to different periods of practice. The editor is not designed for

B or C purposes, and so it does not offer any specific features for achieving such goals. However, the fact that these users have such concerns and tasks to achieve led them to perceive these functional values and suggest related improvements for the system (e.g., add a notepad to document how the session was enacted by the students or make it possible to manage scenario patterns). These suggestions are not improvements of the editor as such; they are improvements of the editor when given these functional values. Similarly, the student-teachers' unexpected functional values illustrate their particular work practice concerns. Functional values F and G are issues that these inexperienced teachers find hard to manage. Given these concerns, the fact that the editor can be used to represent classroom preparations as digital constructions and may therefore potentially be used to implement automated analyses is perceived as a way to achieve constraint-checking tasks. This is not what the system was designed for. In another study involving university lecturers, the editor was perceived as useful for reflecting on scenarios but, more important, as a means to automatically deploy the design of learning management systems such as Moodle (https://moodle.org/) without the related technical burden.

One of the K-12 teachers mentioned only half jokingly that the tool was great for producing tangible proof of thorough classroom preparation for one's supervisor (D). Mentioning this comment to other teachers in other contexts suggested that, although not politically correct, it was not without merit. Typically, given a functional value such as this, the system may be used to document productions constructed elsewhere rather than to conceptualize the teaching setting in a way that is structured by the interface notions, which is its raison d'être. In such a case, the conceptual support that the scenario editor is supposed to provide may become less important than features related to exporting nicely presented scenarios.

2.2. Framing the Notion of Functional Value in an Activity-Centric Perspective

Differentiating Artifacts and Instruments

We frame our perspective in activity theory (AT; see Kaptelinin & Nardi, 2006, or Bødker & Klokmose, 2011, for HCI-oriented presentations), which suggests an activity-centric perspective of how users use artifacts. AT argues that human development and psychological functioning are rooted in social processes and must be addressed in light of the historical development of the individual in society. The relationships between humans and the world are mediated by social and culturally constructed objects. Artifacts are mobilized by users in the context of the activities they consider. They become instruments for users in the context of these activities, when and through the way they allow users to achieve the tasks that users consider in the way they consider them.

Let us consider the scenario editor. The goal of a teacher is not to use the editor but to solve different educational problems related to preparing the class. Teachers have a way of addressing these problems that predates the introduction of

any software. They have developed experience and idiosyncratic practices. An editor, as such, is only one of the many conceptual and technical resources they may use. Their perception of the editor is thus related to the precise tasks they consider and how the system may help them to achieve these tasks given their personal representations, work practices, and context. Software design (e.g., the notions offered as conceptual support—such as "activities" or "groups"—and the associated organizational means) may have an impact on teachers' conceptualizations and work practices. However, it would be rather technocentric to believe that providing teachers such a representation device directly leads them to develop a particular conceptualization or practice.

Such a general perspective leads to the idea that, when considering appropriation, the key notion is not that of artifact but that of instrument.

Implication in Terms of General Perspective

Focusing on functional values and instruments leads to the consideration that appropriation and technical adaptation of software by users are related to those users' efforts to turn software into instruments. Users develop activities in relation to the tasks they consider (e.g., classroom preparation). They consider software (e.g., the editor) as a possible mediator of their activities and may attribute one or more functional values to it. The functional values considered by users may correspond to the purpose for which the system was designed or may not (e.g., editing a scenario or having a tangible record of what has been done in the classroom, respectively). The way users turn software artifacts into instruments may thus lead them to use the system in an unexpected way (e.g., using empty cells to document events or decisions). Moreover, the way users turn the system into an instrument is related to how they usually achieve the underlying tasks. Thus, even when considering tasks that do in fact correspond to the tasks for which the software was designed, factors such as work practices and psychological and/or social constructions may still lead users to consider these tasks in an unexpected way. Therefore, whatever the artifact's design, users may develop unexpected uses, which correspond to the observable part of unexpected instruments. Users' appropriation processes both aim at and require improvement of the activity/software fit. This may call for the system to be adapted, if such means exist.

Figure 2 illustrates the fact that seemingly similar users may perceive different functional values. These values denote the way these teachers see the system after the tests, which would influence their initial usage. However, through effective usage, perceptions will evolve (these functional values will be confirmed or not, others will emerge), and the system will or will not become an effective instrument related to these tasks. Instruments are dynamic entities, and the activity/software fit cannot be considered statically (see Section 3.3).

Designers do not create instruments. Designers create artifacts, which are technological affordances. Instruments are created by users. Informing designing for appropriation thus requires an understanding of how users attribute functional values to software and develop instruments from artifacts, and which elements play a role. We develop such a theoretical account in the next section.

3. THEORETICAL CONSTRUCTION: SOFTWARE APPROPRIATION AS THE DEVELOPMENT OF INSTRUMENTS

In this section, we first study how instruments develop. We then outline two core influencing factors: the fact that users act within ecologies of artifacts, and the fact they are engaged in collective and work practices. This leads us to argue that instruments and systems of instruments should be addressed as dynamic entities. Finally, we outline the general implications for design that arise from this analysis, which will lead to the conceptualization presented in Section 4.

3.1. How Instruments Develop: An Instrumental Perspective

The Instrumental Genesis Theory

To conceptualize how instruments develop, we use instrumental genesis theory (Rabardel, 2003; Rabardel & Waern, 2003). This psychological and activity-oriented theory acknowledges the artifact/instrument dissociation in AT and focuses on the construction of instruments.

In AT, the focus is on the activity, and the instrument is seen as the amplifier of thought. Taking a different perspective, instrumental genesis theory focuses on the process by which humans turn artifacts into instruments, that is, into means to achieve the goals they consider (Rabardel, 2001, 2003). Because artifacts have a mediating role, the development of instruments cannot be addressed by considering just the technical properties of artifacts. An instrument must be seen as an abstract notion combining a technical dimension (the artifact) and a psychological dimension.

Rabardel addressed the psychological dimension of instruments through the notion of usage schemes. A scheme is an invariant organization of behavior for a certain class of situations, or in other words a more or less stable way of addressing specific situations or tasks within which technical and conceptual aspects are intertwined (Vergnaud, 1998). The observable part of mental schemes is the more or less stable sequence of interactions between the user and the artifact for a particular goal (Vergnaud, 1998). Rabardel proposed considering the development of instruments—instrumental genesis—as a dual process comprising instrumentation (i.e., the user's adaptation to the artifact's constraints) and instrumentalization (i.e., the attribution of functions to the artifact and the technical transformation of the artifacts by the user; Rabardel, 2001, 2003). Instrumentation relates to the genesis of the human side of the instrument by developing, adapting, or reorganizing conceptualizations and schemes. Instrumentalization is based on the artifact's initial attributes and properties but extends the initial design by new local or lasting properties in accordance with the current action and situation (Béguin, 2007; Folcher, 2003). Users' activities are seen as both productive and constructive: Users consider tasks and attempt to accomplish them (productive dimension); dialectically, they elaborate resources such as competencies or conceptualizations (constructive dimension). Issues arising while performing a

task may lead users to reconsider and/or develop new resources that, in turn, influence how productive activities are conducted (Folcher, 2003; Rabardel, 2001, 2003).

Instrumental genesis is a psychologically oriented explanation of the process within which users both adapt to artifact constraints and adapt the artifact. Its specific interest is to propose conceptual tools to study this phenomenon. As already mentioned, such a dual process is also referred to in works stemming from other theoretical perspectives (e.g., Mackay, 1990).

Implications of a Developmental Perspective to Instruments

Taking such a developmental perspective leads to the consideration that the way users perceive the functional values of artifacts and turn these artifacts into instruments has two important properties. First, it is related to the conceptualizations and practices developed by these users. Second, it evolves, that is, instruments are dynamic entities.

In this section we illustrate the first aspect with the scenario editor. We come back to the second aspect, which is central to our work, in Section 3.3 (before, we need to introduce aspects related to ecologies of artifacts and collective and work practice aspects; see Section 3.2).

With respect to adaptation means, the distinguishing feature of the editor is that it does not impose a unique representation structure. Unlike traditional instructional languages that impose a given conceptual perspective, the notions used by the editor (activity, group, participant, resource, and role) can be put in different orders, using the left and right arrows at the top of the column (see Figure 1). This feature allows teachers to adopt the structure that best fits their perspective. For instance, some teachers adopt an Activity-Group-Resources representation (i.e., represent a scenario as "a set of activities to be performed by groups sharing some resources"), whereas others use a Participant-Role-Activity-Resource representation (i.e., represent a scenario as "a set of participants playing roles associated with activities and resources"). Let us now consider the following episode. One of the K-12 teachers, while modeling, attempted to merge two cells when this was not possible (having defined two groups as two rows and realizing they had identical resources, she wanted to merge the resource cell; the editor does not accept this because, in some cases, it creates ambiguous constructions). The teacher realized she could easily and neatly solve the problem by changing the column/notions order. She explicitly decided not to do so and kept two separate cells with duplicate values (which was a less neat way of solving the problem). Her explanation was, "I prefer viewing scenarios this way" (i.e., within the conceptualization corresponding to the representation structure she was using). As a similar example, another teacher straightforwardly arranged the conceptual notions in a certain order while commenting, "I always [model scenarios] this way."

These examples illustrate how the users' ways of perceiving and using the system are influenced by previously constructed ways of performing their tasks (in other words: previous instrumental geneses). In these two cases, the system was flexible enough for the teachers to stick to their idiosyncratic practices (though, in the first

case, not flexible enough to allow the teacher to obtain the exact representation she wanted). Actually, even if the editor did not allow teachers to adapt the order of the conceptual notions, it is very likely that teachers would still declare it to be usable. However, in practice, if offered the flexibility to do so, they stick to their idiosyncratic perspective, that is, attempt to do something as they are used to doing it. Mackay's empirical studies highlighted this by showing that when software is upgraded, some users tend to adapt it to "make the new version operate like the old" (Mackay, 1990).

Taking a developmental perspective to instruments thus suggests studying the factors that may influence the origin and development of users' practices. We see these aspects as complex phenomena, that is, as involving different processes interacting in a systemic way, possibly varying from subject to subject and context to context, and that cannot be reduced to any one of these processes. Therefore, claiming that one can identify all factors and understand how they interact may be hazardous. However, acknowledging complexity and the fact that usages cannot be fully predicted does not mean that analyses are useless and, consequently, any design is as good as any other. We believe that attempting to understand and address this complexity is an interesting scientific challenge and helps practitioners both avoid naïve prescriptive views and consider design options in a more informed way.

In the next section, we study three aspects we see as of core importance. First, how users interact with one artifact relates to how they interact with others. Considering software appropriation requires considering users' ecologies of artifacts. The second and third aspects are intertwined: collective and work practices. Considering these three aspects is in line with AT and the instrumental genesis perspectives, as well as with other works related to how users use artifacts, which we summarize in this section. Once these elements have been introduced, we can delve further into the aforementioned second implication of a developmental perspective on instruments, that is, the dynamicity of instruments (see Section 3.3).

3.2. Ecologies of Artifacts, Collective and Work Practice Aspects

Ecologies of Artifacts Aspects

The use of an artifact cannot be understood by focusing solely on the artifact: Users use different artifacts, and these usages influence each other (Bertelsen & Bødker, 2002). This issue may be addressed using the general notion of "personal ecology of interactive artifacts" as introduced by Jung et al. (2008), that is, the set of interactive artifacts that a person owns and/or has access to and uses. See Kaptelinin and Bannon (2012) for a review of different works and a discussion.

Let us start by considering this notion as such. First, a personal ecology of artifacts is not partitioned: Features of different artifacts may overlap and/or interoperate. Second, such ecologies are dynamic: Some artifacts may be or become frequently/infrequently used; new artifacts may enter (or not) for a variety of reasons including functional and nonfunctional aspects, for example, social norms; other

artifacts may not be used anymore and exit. Bødker and Klokmose (2012) highlighted the importance of considering such phenomena, particularly for artifacts sharing features with others.

Let us now take an instrumental perspective. First, one may introduce the more precise notion of an "instrumental cluster of artifacts" to study the subset of artifacts considered by a user for a given task or set of related tasks. For instance, with regard to scenarios, a teacher's cluster of artifacts may include items such as a specific editor, a word processor, and sheets of paper; other items in the ecology of artifacts, such as squares and compasses, do not play any role in the development of the instruments related to managing scenarios. (We use the notion of "cluster" when we need to be more precise, and refer to an "ecology" when speaking more generally.) Second, considering ecologies of artifacts leads to the notion of systems of instruments, that is, organized set of means (Rabardel, 2003).

Considering such instrumental clusters of artifacts, a core aspect is that the artifact–instrument relation is not to be thought of as a 1:1 relation. One artifact may be turned into several instruments (see Figure 2). One instrument may develop from the coordinated use of different artifacts, technically interoperated or not. Different instruments may develop concomitantly from one or several artifacts. An artifact may not be turned into any instrument.

Let us illustrate this with the editor example. Most student-teachers perceived the editor's main functional value as being a means to reflect on and edit the general structure of classroom activities. In line with this perception and their actual work practices, several immediately expressed a need for a new technical feature: print the design in a way that captures both the classroom preparation summary (the table) and the textual details (the open text description of the different items, which is accessed by double-clicking on them). Their rationale was that it is simply not possible to keep an eye on the editor during classroom sessions. In direct contrast, a printed version is easy to consult and to annotate. Consequently, the editor has a useful preparation-support functional value if and only if it can be interoperated with another artifact (in this case, a sheet of paper). The academics had a different request: interoperate the table with the means that learning management systems offer to supervise students' activities (i.e., means to check how the learning activities designed via the editor are implemented by students). In other words, the use of the editor will develop in relation to its features, its perceived functional values, and other preexisting artifacts and their functional values (and vice versa for the use of these other artifacts).

Software appropriation (and, thus, how to design software for appropriation) therefore cannot be addressed by considering artifacts in isolation. Use of a new system such as the editor must be regarded as a modification of the instrumental cluster of artifacts that teachers use to mediate the activity of preparing learning sessions and, more generally, their professional activities (teaching, reporting, or exploring new teaching means). Appropriation will develop from how teachers take advantage of different artifacts, including the editor, to carry out their work practices.

Collective and Work Practice Aspects

The fact that software adaptation cannot be considered a primarily individual activity has been highlighted by different works such as Mackay's studies (Mackay, 1990). These studies showed how users share adaptations, with some actors acting as leaders and helping others. This collective dimension was further confirmed by other studies such as (Gantt & Nardi, 1992), which analyzed how users collaborate with "more experienced users" to adapt software to their needs. In line with this perspective, Trigg and Bødker (1994) emphasized that software adaptation is not to be seen as individuals just solving their individual usability issues. It may play a role in organization-wide efforts to standardize procedures as emergent solutions of situated problems.

In these different works, the collective dimension is intrinsically related to another aspect: the role of work practices. This is also central in the instrumental genesis theory, which is rooted in ergonomics and professional didactics (Rabardel, 2001, 2003). Instrumentation and instrumentalization processes are closely related to the professional dimension of the activity and how individual processes join collective practices (Gueudet & Trouche, 2011).

Another perspective that sheds similar light on the importance of collective and work practice aspects is the genre approach, which builds on Bakhtin's notion of genre in rhetorical and literary analyses to consider typified social actions. Genres are "socially recognized types of communicative actions used by organizational members for particular communicative and collaborative purposes" (Yates & Orlikowski, 2002, p. 14). A genre (e.g., project proposals, annotations, reports) typically involves different media, artifacts, structural features, and linguistic features. As traditions of using tools, genres implicitly convey a worldview and allow subjects to recognize the activity and the appropriate actions (Russell, 2009; Spinuzzi & Zachry, 2000). They may be seen as the tacit part of the activity, which is known and shared by the workers of a given setting and defines what they are expected to do without having to respecify tasks each time they appear (Clot & Faïta, 2000). With respect to appropriation, this means that how users perceive and use artifacts find explanations in the characteristics of the professional activity. As complex activity systems involve different artifacts, the genre approach acknowledges the notion of ecology of artifacts or "genre ecologies" (Spinuzzi & Zachry, 2000; Yates & Orlikowski, 2002). As other analytical frameworks such as contextual design's work models or distributed cognition's functional systems, it may be used to address what Spinuzzi (2001) called compound mediation, that is, how workers coordinate artifacts to help them get their jobs done. Yates and Orlikowski (2002, p. 32) noted that "in moving their communication to a new medium, members of a team or community typically import existing genres and genre systems, improvise around them, and gradually learn to take advantage of new opportunities afforded by the medium." As one can see, these different considerations are consistent with the perspective we develop in this article.

3.3. Dynamicity of Instruments and Systems of Instruments

Having outlined the importance of ecologies of artifacts and collective and work practices, we can come back in more detail to the second implication of adopting a developmental perspective of instruments, that is, the dynamicity of instruments and systems of instruments.

Considering developmental and ecology of artifacts perspectives together helps in understanding that users may develop new instruments in relation to both new and already-used artifacts. Users' instrumental clusters of artifacts evolve both from the outside, when new artifacts are offered to or imposed on users, and from the inside, when users actively decide to use (or change their use of) artifacts in relation to a task. These two evolutions may interrelate in different ways. Typically, the evolution of personal or collective work practices, motivations, and/or skills may cause the perceived functional values of existing artifacts to evolve. Old artifacts may be given new functional values or stripped of previous functional values independently of and/or in connection with the (new) functional values of other (old or new) artifacts, and as a consequence, users may develop new instruments (or new versions of instruments) from these artifacts. The evolution of utilization schemes and/or artifact transformations may also cause instruments to evolve. This may occur in relation to the evolution of an attributed functional value or not, that is, the functional value is unchanged, but the way users take advantage of the artifact to achieve their goals evolves. Finally, users' systems of instruments may evolve in relation to the developed instruments (i.e., instruments appear, disappear, and evolve) and/or in relation to the way users relate these instruments to each other (i.e., an evolution as a system).

Let us present examples. The way some teachers suggested using the editor to exchange scenarios and share pedagogical ideas illustrates a possible collective evolution. An example of the "system" aspects is as follows: Teachers' standard preparation sheets are often structured as tables edited in word processors. This outcome of teacher training and past instrumental geneses may be a key factor for teachers to adopt a new artifact (the scenario editor) that allows them to continue their way of acting with new means offering added value. Because teachers use word processors for a range of tasks, it is unlikely that this device will exit the ecology, but its functional value for editing scenarios may be diminished or lost. However, this may be the starting point for another usage of word processors. For example, several K-12 teachers suggested that the editor should be enhanced with means to manage taxonomies of inspiring activity verbs (e.g., "present," "analyze," or "summarize"). This demand, which arises from the use of the editor, cannot be handled with the present system. It could be the seed for giving the word processor the new functional value of managing the data that will be used with the scenario editor. In other words, the word processor appropriation process does not end when the scenario editor appears; it continues in a way that may be affected by the introduction and appropriation of this new artifact. In turn, the initial appropriation of the new artifact is affected by how the old artifact has already been appropriated (e.g., how teachers manipulate tables in word processors) and how appropriation of this old artifact evolves. All these aspects are framed and influenced by the overall work practice.

3.4. Synthesis and Implications for Conceptualizing Design for Appropriation

In Section 2, we stated that the appropriation and technical adaptation of software are related to users' efforts to turn software into instruments. The theoretical account we developed in this section allows us to be more precise. Users develop instruments by interacting with the tasks they consider within their ecology of artifacts. These interactions take place within personal and professional development processes, and in sociotechnical contexts: Users have their personal background, knowledge, professional practices, and motivations; they are members of communities; they act in situated contexts. All these considerations are related to broader (society, culture) aspects. Through practice, users may develop new instruments and systems of instruments from both artifacts entering their ecology and already-present artifacts, and develop successive systems of instruments in relation to the development of new instruments, the shaping/abandon of others, or the evolution of how they are related one to the other.

The first implication of this understanding is that the consistency of the designed features of a system with users' effective activities cannot be considered statically. Activity is both productive, that is, oriented toward the production of results, and constructive, that is, oriented toward the development of the subject's instruments and resources; this is an ontological characteristic of human activity (Rabardel & Bourmaud, 2003). As a consequence, users may turn the same artifact (and, furthermore, groups of artifacts) into different instruments or systems of instruments over time, and features at one time appropriate for users' activities may, later on, only partially support them or even become obstacles. Therefore, even if looking at a single user and a perfect requirements analysis, artifact/instrument inconsistencies may still occur in relation to the developmental aspects of practices. More generally, different users may develop different instruments from the same artifacts, as the conceptualizations and/or idiosyncratic practices developed by these individuals or local groups may differ from what is considered "standard" within an institution or a community. And, of course, different users have different ecologies of artifacts. This leads to a clear-cut conclusion: UCD, if it does not acknowledge and address appropriation processes, has intrinsic limitations. We come back to this in Section 6.

The second implication is that, to efficiently transform artifacts into instruments in practice, users should be empowered to adapt the designer's initial design or, in other words, to continue design in use (Rabardel, 2003). Instrumentalization processes suggest that users are assumed to be designers in the very real sense of this word (Kaptelinin, 2003; Kaptelinin & Nardi, 2006).

This analysis thus leads to the view introduced in Section 1: Designing for appropriation should be thought of as providing users with means to adapt the artifacts they use and align them with their personal and/or collective developed/developing instruments. It should be thought of as a strategy to offer users means to solve their instruments/artifact tensions themselves.

Empowering users to continue design in use raises both theoretical and practical issues (Kaptelinin & Nardi, 2006). The first question is to clarify what "design by users" really corresponds to and how it relates to "design by designers." The second question is to help designers implement the appealing but general idea of providing users the means to continue the design in use. We examine the first question in Section 4 and the second in Section 5.

4. A CONCEPTUALIZATION: "DESIGN FOR APPROPRIATION" AS EMPOWERING USERS TO CONTINUE SOFTWARE DESIGN IN USE

In this section, we propose a clarification of what design by users may correspond to and how it relates to design by designers. First, we propose a general framework. Second, we distinguish three perspectives on how users may act as designers: usage-informed design, community-of-users design, and user adaptation. Finally, we analyze user adaptation in more detail.

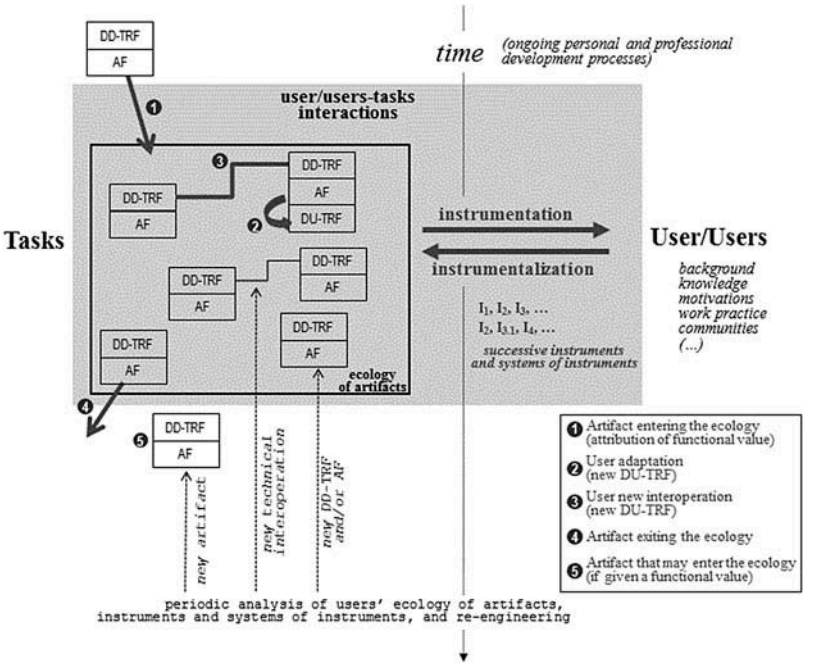
4.1. General Framework

A distinctive characteristic of software is that the artifact (the piece of code) may include task-related features (i.e., means for users to achieve the task), as well as means for users to adapt these task-related features or develop new ones. In other words, users may develop instruments from both a psychological perspective and a technical perspective: development of new ways of using an existing system; integration of a new system in the ecology; adaptation of an existing system; or establishment of new interoperability, which may develop technically and/or functionally, for example, by using these systems in an organized way. Software adaptations may thus be implemented and new interoperability established by designers, via usage analyses and reengineering, and/or by users who exploit the system's adaptation features.

The general framework we propose for conceptualizing these different cases is as follows (see Figure 3): Users, as individuals and as groups, interact with the task(s) they consider via different media, including software artifacts (e.g., the scenario editor). These software artifacts present Designed-by-Designers Task-Related Features—that is, features that were designed by designers to enable users to achieve the tasks for which the system has been designed (e.g., the feature "assigning a student to a group")—and Adaptation Features. While interacting with tasks, and as part of their ongoing personal and professional developmental processes, users develop successive instruments and systems of instruments via instrumentation and instrumentalization processes. Within these processes, they may develop Designed-by-Users Task-Related Features by adapting artifacts (using their Adaptation Features) or interoperating them.

In this framework, Designed-by-Users Task-Related Features are to be thought of in a conceptual sense. They may correspond to, and be reified as, features in the

FIGURE 3. Offering users adaptation means and conducting periodic re-seeding. *Note.* Software artifacts are seen as comprising Designed-by-Designers Task-Related Features (DD-TRFs), Adaptation Features (AFs) and, potentially, Designed-by-Users Task-Related Features (DU-TRFs).



traditional sense, for example, teachers creating a macro to check scenario constraints such as whether all students are engaged in individual and collective activities. They may also correspond to particular ways of using the system, for example, saving empty scenario structures as a way to model from patterns. Similarly, interoperability may develop functionally, for example, use of the word processor to manage taxonomies of action verbs. Although not EUD in the traditional sense, this nevertheless corresponds to users developing new instruments.

4.2. Three Perspectives on How Users May Continue Design in Use

Referring to this general framework, we distinguish different interrelated cases of design continued in use (see Figure 4 for a summary, which also includes aspects discussed in Section 4.3).

Usage-Informed Design

Usage-informed design is design continued in use as the continuous reengineering of the system according to effective usages. This corresponds to the "periodic analysis of users' ecologies of artifacts, instruments and systems of instruments, and

FIGURE 4. Design continued in use: Different perspectives.

		Control of the Adaptation/ Evolution				
		Design Decisions	*Technical Implementation*	Time Span	Focus	General Objective
Usage-informed design perspective		Designers	Professional developers	Refinements via re-engineering iterations	Design issues highlighted by effective usages that differ from expectations	Improve the design to fit users' needs and usages
Community-of-users design perspective		Users	Users or professional developers	Evolution	Evolution of the perspective of the community's needs and/or practices	Allow the system to evolve with the community's perspectives and practices
User-adaptation perspective	Alignment episode	User	User	Alignment via persistent adaptations	Individual practices	Allow individuals to adapt artifacts to their existing idiosyncratic practices
	Instrumentalization episode	User	User	Evolution via persistent adaptations	Individual practices	Allow individuals to adapt artifacts within the evolution of their practices
	Local episode	User	User	Punctual episode	Individual local needs	Allow individuals to adapt artifacts to a particular setting

re-engineering" path in Figure 3. Users are provided with new or more suitable (Designed-by-Designers) Task-Related Features via new versions of existing artifacts, new interoperations, and/or new artifacts. New artifacts will enter users' ecologies of artifacts and, more precisely, instrumental clusters of artifacts, if and as they are given functional values. Although usage-informed design may be technically implemented by professional developers, it can be said that users have adapted software to their needs in that they influenced the design via their usages.

An illustration is as follows. Given that teachers editing a scenario seem to start by listing the items—activities, groups, or resources—and then organizing them within the table, the design could be adapted to make it easy to define sets of activities or groups. And, if it is confirmed that teachers use the editor in conjunction with a word processor, features to manage data in compatible formats could be offered.

When considering the usage-informed design perspective, matters of concern relate to how coordinated analyses of users' ecologies of artifacts, instruments, and systems of instruments, and the evolution of these elements, may inform design iterations. As highlighted by Béguin (2003, 2007), if instrumental genesis is a response to initial design, designers may also respond to instrumentalization processes. We review some possible means of analysis in Section 5.2.

User Adaptation/Design

User adaptation is design continued in use as the adaptation of software by users as individuals. The user, as an active and intentional subject, adapts software to his or her practice and develops his or her own (Designed-by-Users) Task-Related Features, which may include technical modifications using the available Adaptation Features.

An illustration is as follows. As principles underlying scenarios can often be reused in different domains, some teachers questioned whether the editor could be used to manage such patterns. This is not the case. However, given a scenario with a reusable structure, such a feature may be approached by removing all the items (i.e., the particular activities or groups) and saving the "empty" table. In the case of a jigsaw scenario, this provides a pattern (an initial individual activity followed by rows to separate groups and then mix them again) from which the teacher may edit instances. With the current editor, such empty constructions must be saved as scenarios; they cannot be managed separately, as the "pattern" notion does not exist. However, if the editor offered secondary notation systems such as allowing users to manage filename extensions, teachers could use them to design (to "user-design") a new pattern-management task-related feature.

When considering this user-adaptation perspective, matters of concern relate to how individual users may be empowered to change or extend some of the software features according to their effective practices and needs. As introduced in Section 1, the issue is twofold. On the technical side, the question relates to the implementation of Adaptation Features, typically as tailorability/customization means. On the non-technical side, the problem is how to identify which Adaptation Features to offer, that is, which aspects users should be able to adapt or extend and how.

With respect to appropriation, however, user adaptations may correspond to different realities. We further examine this question in Section 4.3.

Community-of-Users Design

Community-of-users design is design continued in use by a community of users. As in the case of the usage-informed design perspective, it corresponds to the "analysis and reengineering" path in Figure 3. Implementation, however, may combine a number of realities: one-off development of features by information and communication technologies-competent users using the Adaptation Features provided by designers or going into the code, organized developments by an established community of practice, or development by professional engineers based on emergent requirements. With respect to design by designers, such reengineering is usually not structured into a rigorous process as it is within the usage-informed design perspective, and may rather be seen as synthetic interpretations of multiple emergent actions.

An illustration is as follows. Different works have studied how teachers share resources via communities of practice; see, for instance, the analysis of how instrumental processes underlie how an association of mathematics teachers shaped a set of educational resources (Gueudet & Trouche, 2009, 2011). Let us suppose such a community of practice emerges for the scenario editor, with teachers using it to share scenarios both on a one-to-one basis and by building a repository. In response to this emergent use, the community may manage to shape the editor and develop (Designed-by-Users) collaborative features. This may be achieved in different ways. One option is to technically adapt the editor (e.g., going into the code to implement new means to index resources). Another is to connect the editor with other artifacts (already part of the community's ecology of artifacts or entering the ecology for this purpose; e.g., forums or wikis).

When considering this community-of-users perspective, matters of concern relate to the community dimension, the way the artifacts are progressively shaped by this community, and the means that may be provided to this community.

4.3. Understanding Users' Individual Adaptations

We stated that users individually adapt software to make it compatible with the functional values they attribute to it. Does this mean that every adaptation action is to be understood as an "instrumentalization action," as a symptom of the development of an instrument? We do not think so. Understanding users' individual adaptations requires deeper analysis.

Developmental Processes Are Lengthy and Complex

Relating users' individual adaptations to instrumentalization processes is inspiring, but it sets up different theoretical traps.

First, there is an issue related to time. Technical adaptations of software take place as episodes: At a given moment, users take advantage of some adaptation means to adapt the system. In direct contrast, instrumentalization is not to be thought of as a particular moment. It builds on instrumentation, and these processes develop in a dual way. Users' adaptations of artifacts in relation to instrumentalization are an output of these processes and, more generally, the long-term developmental process of which they are a part. As an example, studies on calculators (Trouche, 2004) highlighted that technical adaptations such as creating keyboard shortcuts or storing formulas were an outcome of different development stages, from the discovery of the artifact and its relevant keys to making it fit one's hand by transforming it. It would thus be misleading to interpret every single user adaptation as an "instrumentalization action."

Second, the actions of users taking advantage of adaptation means (e.g., to adapt the order of the conceptual notions in the editor) may present different, possibly simultaneous, dimensions. First, they may have a situated dimension. Adaptations are considered in the context of a given setting involving a specific task, context, and moment. They are "here and now." As emphasized by situated cognition theory, situations have a determining influence on activity. The teacher considers a given scenario, realizes it is not easy to enact with the Activity–Group–Participants–Resource representation structure, and decides to change to a Role–Participants–Resource structure. Second, adaptations may have an interpretative dimension. This may arise in relation to recognition of a class of settings and preexisting schemes (Vergnaud, 1998). Teachers' oral comments such as "I always [model scenarios] this way" likely denote invariants developed as part of professional development. The setting is interpreted in the light of preexisting constructions that have crystallized. Third, adaptations may have a constructive dimension and be part of a more general process that takes place within the development or refinement of schemes (Kaptelinin & Nardi, 2006; Rabardel, 2003). Finally, adaptations may have an oriented dimension. Adaptation is related to motivations, and users may have very different motivations (see Figure 2).

Individual Software Adaptations: Different Realities

The different considerations introduced here suggest that individual software adaptations may correspond to different realities, which we propose to clarify with the following distinctions:

1. Alignment episode. Adaptation is related to how users use adaptation features to make software consistent and operational with preexisting constructions (e.g., representations or organized ways of acting developed through previous instrumental geneses).
2. Instrumentalization episode. Adaptation takes place within an ongoing instrumental genesis.
3. Local episode. Adaptation is contingent, incidental, not significantly related to a preexisting or underdevelopment specific organization of action.

The distinction we introduced between alignment and instrumentalization episodes, that is, adaptations as related to past versus ongoing geneses, is analytical. Users' crystallized organizations of action are baselines, but they also evolve.

Let us illustrate this complexity with the teacher facing a technical constraint that did not allow her to model the scenario as she wanted. Although aware of how to solve the problem by changing the order of the notions, she opted not to do so because she has *her* way of viewing scenarios, wants to use the editor in a way consistent with this preference, and thus aligns it with her view. See also other teachers' comments such as, "I always [model scenarios] this way" or "I'd like to keep my words" [referring to the offered notions], or empirical studies showing how some users adapt software to "make the new version operate like the old" (Mackay, 1990). Through their work practices, these teachers have developed idiosyncratic perspectives and want to stick to them when using the editor, which, in some cases, requires using the system's adaptation means. Instruments, however, are not developed at this particular moment: Adaptation is used to avoid obstacles to the outputs of previous developmental processes. Nevertheless, such alignment episodes may also be part of a more general instrumentalization process; that is, to be more precise, adaptation may be used to avoid obstacles to the current outputs of an ongoing developmental process. Similarly, adaptations appearing incidental when considered as such may, taking a more general perspective, be part of an exploration of software features and an informal trial-and-error process that will retrospectively appear to be part of the construction of something. Nevertheless, some incidental adaptations may persist though not corresponding to any symptom of appropriation (e.g., the adaptation is not relevant but does not create more interference than the initial structure, and software is not modified back). Interpreting user adaptations as alignment or instrumentalization episodes requires an analysis of the user as an active subject, and over the long term.

4.4. Discussion

In this section, we clarified "design by users" by distinguishing usage-informed design, community-of-users design, and user adaptation. According to the adopted perspective, empowering users to continue design in use corresponds to different realities: using adapted means of analysis to inform designers and help them to match users' needs and usages, allowing a community to make a system evolve, and allowing individuals to adapt artifacts to their particular practices. In the next section, we study the analysis, software engineering, and technical means that may be considered.

The three perspectives we distinguished are not contradictory. The differences are conceptual and relate to the level and focus of the analysis. What actually happens may be a complex hybrid pragmatic process. Users' individual adaptations may take place within a community and nurture community development, which in turn may inform professional developers' design or redesign.

The notions of alignment and instrumentalization episodes, which we introduced to clarify user adaptation, may also inform the analysis of changes in ecologies of

artifacts. Unexpected use and/or adaptation of a new artifact may be because it is "aligned" with existing schemes developed in interaction with an old artifact. In direct contrast, smart transitions may be explained by the fact that the user developed usages and schemes that the old artifact inhibited and the new one fits, a case for which instrumentalization episodes of the old artifact may appear as symptoms. Such analyses may thus have a heuristic value for analyzing the functional values that users attribute to systems and, on this basis, for considering what means of adaptation may be offered.

5. SYNTHESIS, IMPLICATIONS FOR DESIGN, MEANS, AND OPEN QUESTIONS FOR FUTURE RESEARCH

In this section, as a takeaway for designers and theorists, we develop the value of insights presented in Sections 2, 3, and 4 in terms of implications for design and of new areas of study. We successively present a summary of the perspective and the general principles it suggests; the methodological means that may be used to conduct the required analyses; the software engineering and technical means; and, finally, some research questions that this perspective raises. Section 6 proposes a discussion including similarities to and differences from other design perspectives.

5.1. Designing for Appropriation: Conceptualization and Principles

As a conceptual basis for designers to become prepared for action, the theoretical considerations we have developed may be summarized as follows: Appropriation of software is related to the way users, while interacting with the tasks they consider, attribute functional values to software artifacts and, by considering them as mediators of their activities, turn them into instruments for themselves. This process is influenced by preexisting psychological and social constructions and their evolution, including aspects related to work practices. It may lead to native or developing tensions between the instruments developed by users and the existing artifacts' features, which may require/lead users to adapt software. As the development of instruments is an expression of the ontological constructive nature of human activity, these artifacts/usage tensions are not to be thought of as related solely to "good" or "bad" design, and as a one-shot issue. The fact that users continue design in use is not an issue to be solved, but a phenomenon to be taken into account.

This conceptual basis suggests the following general principles:

1. Proposing and reconsidering means of adaptation should be an explicit concern in artifact design cycles. Professional designers may periodically adapt software to align it with usages. However, this will not stop usages from developing. Reseeding the system should thus be thought of as considering two objectives: first, offering professionally addressed task-related features that implement and potentially

enhance those developed (in a conceptual sense and/or via effective technical adaptations) by users themselves; second, offering new adaptation means.
2. Specifying the adaptation means to be offered to users must be based on the identification of (or hypotheses related to) the instruments that users have developed and are developing, that is, to how and why users take advantage of the artifact while interacting with the effective tasks they consider.
3. Studying users' instruments requires considering the functional values they attribute to software, their ecologies of artifacts, and the psychological and social constructions related to these artifacts. It must be acknowledged that an instrument may build on several artifacts, that artifacts may be part of several instruments, that users consider systems of instruments, and that instruments and systems of instruments develop.
4. Offering means for users to deal with the technical (artifacts') aspects of shaping their instruments must be considered at both a macrolevel (enabling and supporting community-of-users design) and a microlevel (enabling and supporting individuals' adaptations of systems). Users develop both as individuals and as communities, and within their development, individuals and communities engage in appropriation processes.
5. Means of adaptation should be offered in a way that encourages and supports users (as individuals and as communities) in analyzing, at a more abstract level than immediate needs, how the artifact's features fit with their activities, and how adaptations may improve this fit.

A general life cycle (focusing on one artifact only for the sake of simplicity), is as follows:

1. Design of an artifact providing initial task-related features and a priori adaptation features. This artifact's requirements analysis must consider users' present ecology of artifacts, instruments, and systems of instruments.
2. Analysis of effective usages, paying specific attention to designed-by-users task-related features in a broad sense (customization, technical adaptations, technical or logical interoperations).
3. Reseed the system, that is, adapt or redesign the system to provide new or adapted task-related features, new means of adaptation, and/or new means of interoperation in line with the way usages developed.

5.2. Review of Analysis-Oriented Methodological Means

Applying the aforementioned principles requires analyzing users' ecologies of artifacts, the functional values users attribute to software, the reasons for this (e.g., work practices), how users seem to turn artifacts in instruments, and how usages develop in relation to these aspects. In this section we propose a review of the possible approaches to such analyses, which is of course not exhaustive. Figure 5 summarizes

FIGURE 5. Objects of study and possible means.

approach and references \ object of study	Functional Values, Development of Instruments	User's Ecology of Artifacts	Influence of Work Practices
Activity theory-oriented works (Kaptelinin & Nardi, 2006); developmental work research methodologies (Kuutti & Syrjänen, 2011)	++	+	+++
Instrumental approach (Rabardel, 2001, 2003)	+++	++	+++
Human-artifact model (Bødker & Klokmose 2011, 2012)	+++	+++	++
Genre approach (Russell, 2009; Spinuzzi & Zachry, 2000; Yates & Orlikowski, 2002)	++	+++	+++
Interconnections of artifacts with personal life (Jung, Stolterman, Ryan, Thompson, & Siegel, 2008; Ryan & al., 2009)	+	+++	

the extent to which the approaches most referred to in this article consider or encourage considering these different objects of study (ranking is indicative of our personal analysis).

Jung et al. (2008) showed how personal ecologies of artifacts may be analyzed through two phases: an inventory phase (identifying the artifacts people own, how they use them, and how they value them) and a mapping phase (visualizing implicit or explicit relationships among artifacts as an ecological map). This process is thought of as a means to help designers "consider and predict the overall influences of digital artifacts in human life." A tool implementing this analysis approach is described in Ryan et al. (2009). These authors raised the importance of considering how users believe that devices are connected. Taking our perspective, this contributes to understanding the functional values perceived by users and what underlies their perceptions.

Spinuzzi (2002) showed how to model ecologies of artifacts by coding genres and characterizing their relationships based on how workers connect them in the course of their work. He demonstrated that such models are helpful in planning and designing (in this case, computer documentation).

Yates and Orlikowski (2002) showed how studying the different dimensions around which genre systems carry expectations ("why," "what," "who/m," "how," "when," and "where") helps understand changes in communicative interactions associated with the adoption and use of artifacts.

Béguin (2003) showed how intermediate outcomes of design help mediate a dialogical process between users and designers and create mutual learning conditions, which is of particular importance when considering usage-informed design.

He shows that when one actor appropriates the contribution designed by others, it changes the way the designed object is experienced by them. More generally, such a dialogical process may be supported by conceptual and methodological resources for multidisciplinary work; as an example, see Tchounikine (2011).

Bødker and Klokmose (2011) showed how the human-artifact model they proposed may be used to frame activity-centered analyses and design. This model introduces the three levels of AT ("why," "what," and "how" considerations) on both the human and the artifact sides. It helps in analyzing what users do ("What do they say they do?" "What is the artifact used for?" and "What may it be used for?") and why (motivational aspects). This is useful for studying the functional values users attribute to software. The authors indicate different methods that may be used to conduct such analyses. Bødker and Klokmose (2012) then showed how the human–artifact model may be used to study the dynamicity of users' ecologies of artifacts. The introduction of new artifacts is analyzed in terms of cycles, including an unsatisfactory state (the artifact ecology no longer fulfills user needs), an excited state (a new artifact is added to the ecology, leading to exploration and reassessment of roles), and a stable state (new and old artifacts have found their role).

Rabardel (2001, 2003) showed how the development of instruments may be analyzed by focusing on the activities that actors must handle professionally. He proposed studying these activities in terms of classes of situations, that is, situations with similar characteristics (e.g., tasks to be performed or aspects to be considered), leading to relatively stable activity modalities, and for which actors associate/develop adapted activity schemes and instruments. Such classes may be grouped in activity families related to a similar general purpose, which may be used as a unit of analysis for the professional domain of activity. Examples of instrumental geneses analyses may be found in Rabardel and Bourmaud (2003), for dispatchers assigning technicians to tasks; Trouche (2003, 2004), for students enhancing symbolic calculators; Cerratto Pargman (2003), for collaborative writing; Folcher (2003), for hotline operators using a shared database; and Gueudet and Trouche (2009), for a community of teachers using documentation systems. To study systems of instruments, Rabardel suggested a "failure and substitution of resources method (p. 681)." The principle is, within a class of situation, to study what happens in the absence of an artifact: what functions are substituted, how, and what the impact on activity is. The functional equivalence among instruments may be analyzed in terms of substitutable functions, substitution conditions, and substitution values.

Engaging users in reflective analysis may be addressed, classically, using practical tools inspired by developmental work research methodologies (Kuutti & Syrjänen, 2011).

In our analysis, we assumed that users were the intended users, whose requirements were taken into account. However, software may also be used by unexpected users who were neither targeted nor anticipated by the designers. Quinones, Teasley, and Lonn (2013) showed how such cases may be studied using a matrix that crosses an anticipated/unanticipated axis with an expected/unexpected user axis.

We mentioned that the notions of alignment and instrumentalization episodes may inform the analysis of changes in ecologies of artifacts' and of the functional values users attribute to systems. Our experience also suggests that users' requests may have a heuristic value. As examples: Teachers' requests such as the opportunity to use "one's own words" rather than the notions provided suggest that the editor is seen as an organizational direct-manipulation device rather than as a conceptual modeling support; requests such as the possibility to add a column to note comments as the scenario unfolds suggest that the editor is seen as a means to draw on experience and reengineer scenarios for future sessions; requests such as the possibility to introduce constraint-checking mechanisms suggest that the editor is seen as a means to help improve the quality and consistency of scenarios.

5.3. Review of Software Engineering and Technical Approaches

From a technical perspective, offering users adaptation features is addressed by EUD, that is, the "methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create, modify or extend a software artifact" (Lieberman, Paterno, Kalnn, & Wulf, 2006, p. 2). EUD comprises different approaches such as tailorability, metadesign, and end-user programming, which intersect in different ways; see Ko et al. (2011) for a review. We first review these approaches with respect to our perspective and then illustrate that offering means of adaptation may only require smart design and basic techniques.

Tailorability, Metadesign, and End-User Programming

Tailorability advocates providing users with integrated support for modifying the system in the usage context (Lieberman et al., 2006; Mørch, 1997; Wulf, Pipek, & Won, 2008). It is originally meant to address the fact that user needs are difficult to identify and traditional development cycles involving designers and implementers are time-consuming and expensive. To clarify the picture, Mørch (1997) introduced a distinction between customization (modifying the system by choosing attribute values from a predefined set), integration (adding new features to the system by linking predefined components together), and extension (adding new code). Integration is very common in web applications, as evidenced by all the add-ons available to enhance browsers and the ease with which users can design web pages by integrating different services. Extension comes close to end-user programming. In both cases, component-based approaches seem promising (Mørch, 2011), as do visual programming and programming by demonstration. However, tailorability issues are not limited to offering users simple interfaces. Difficulties may arise from the gap between the concepts considered by users when using the system and the ones underlying the system implementation (Stevens, Quaisser, & Klann, 2006), or from the necessity to address infrastructural issues (Wulf et al., 2008).

Metadesign is a holistic perspective that provides an inspiring general framework. It advocates creating sociotechnical environments that allow users, as the owners of

problems, to act as designers and close the gap between how system developers imagined and anticipated users' needs and how users actually use the system in the real world (Ye & Fischer, 2007). It extends UCD and participatory design in the sense that users are meant to act throughout the whole system life cycle (Fischer, 2003). The Seeding, Evolutionary growth, and Reseeding model (SER model; Fischer, Giaccardi, Ye, Sutcliffe, & Mehandjiev, 2004) suggests considering three phases: initial professional development proposing a "seed"; usage periods during which users may adapt, complete, or extend the system according to their needs and context (evolutionary growth); and incorporating these changes (reseed). The framework we propose matches this perspective (see the discussion in Section 6).

An important difference between tailoring and metadesign is the way adaptation is introduced (some tailoring-oriented works such as Mørch, 2011, however, come close to the SER model). Stevens et al. (2006, p. 270) considered that "tailoring is carried out when a breakdown situation occurs while using a tool". They referred to empirical studies showing that finding the appropriate tailoring functions is a barrier that either prevents tailoring or adds significantly to its cost. The introduction of a new version of an application or of tasks that require modified functionality is seen as a relevant occasion that may incite users to perform tailoring (Wulf et al., 2008). This, however, is a rather discrete perspective: usage, breakdown or external event, tailoring. Metadesign suggests more continuity (periodic reseeding), with professional developers and end-users acting as a team. Establishing a "culture" within which adapting software is acknowledged to be allowed, easy, and socially acceptable, is of high importance (MacLean, Carter, Lovstrand, & Moran, 1990).

Finally, end-user programming is defined as "programming to achieve the result of a program primarily for personal, rather [than] public, use" (Ko et al., 2011, p. 5). This requires users to be able to program, which is not the most frequent case but may make sense when an appropriate context is created. As an example, on the basis of experience and empirical studies, Costabile, Mussio, Parasiliti Provenza, and Piccinno (2008) argued that domain experts do not want to become programmers as such, but if adaptation is necessary for their intrinsically motivated goal of achieving their tasks, they may overcome potential difficulties and act as unwitting end-user programmers. To support such a process, a metadesign methodology acknowledging the fact that users work in teams and cooperatively construct solutions to their problems is proposed. Users engage in Software Shaping Workshops, enabling them to perform development activities by interacting through a domain-oriented visual language tailored to their culture. As another example, trials to solve the issue of engaging users in end-user programming have been conducted based on the identification of "teachable moments" to provide contextual illustrations, using a Surprise–Explain–Reward strategy (Ko et al., 2011).

Offering Metadesign-Inspired Means of Adaptation Using Basic Techniques

Designing for appropriation principles may also be implemented by smart design informed by the general perspective we have developed and basic techniques such

as customization options, secondary notation systems, abstract implementation of structures, macros, or plug-ins.

Let us take illustrations from the scenario editor. It implements the general metadesign principle of engaging users in design with respect to a limited, though extremely important, element: the order of the conceptual notions used to model scenarios. As shown by tests, teachers take advantage of this feature to adopt the modeling structure that fits their perspective. This is far from being anecdotal. Conceptualizing a scenario as "a set of activities to be realized by groups sharing some resources" is significantly different from seeing it as "a set of participants playing roles associated to activities," and using one representation structure when one's desire is to use the other is problematic. This adaptation feature is, for users, a customization feature. From a modeling and technical perspective, it requires model-based manipulation processes based on the underlying tree model (Sobreira & Tchounikine, 2012). As another example, given inputs such as "I would like to use my words," a generalization within which users can define the notions they use has been implemented using basic abstract-implementation-of-structures techniques (Sobreira & Tchounikine, 2015).

Let us take a final example of adaptation features that allow users to "continue design in use." The tests suggested that different teachers were interested in managing different types of objects, such as verbs that often appear in scenario (e.g., "read," "synthesize," or "solve"), representation structures (e.g., the Activity–Group–Participant–Resource–Role structure), and sets of rows and columns denoting a general organization (e.g., the jigsaw pattern). These different suggestions may be addressed individually. However, taking a more abstract perspective, one may consider the general specification "allowing users to manage the constructions they build." This feature may be implemented by providing means for users to define (save, manage) "constructions" while letting users define what "constructions" they want to manage (e.g. educational notions, representation structures, or scenario patterns) and how they want to organize them. Such an implementation, which is similar to the secondary notation principle, allows for a family of usages to develop.

5.4. New Research Questions

The way we suggest considering designing for appropriation opens a number of interrelated theoretical, methodological, and technical research questions for the HCI research community.

We suggested that identification of the adaptation features to offer should build on analysis of the functional values that users attribute to software, and we listed some potential means of analysis. However, this must be further elaborated. In particular, guiding strategies must be established and methodological issues related to the time span and complexity of instrumental processes addressed. Three important and related aspects, which we mentioned only in passing in this article, must also be considered as primary issues: How does one avoid overwhelming users by offering them means of adaptation from which they can conceptually and technically act productively? How does one offer means of adaptation in a way that allows users to

relate them to their activities and to act without disrupting these activities? How does one engage users in actively considering adaptations? Focusing on instruments also raises the theoretical issue, first mentioned by Kaptelinin and Nardi (2006), of how to characterize the extent to which an artifact may be considered an instrument. The framework we proposed raises the additional question of how to distinguish alignment, instrumentalization, and local episodes. More generally, a deeper understanding of the instrumentation/instrumentalization interplay and the different factors that may play a role (e.g., psychological, social) is needed.

From a technical perspective, in addition to further examining EUD issues within this new framework, the perspective we developed raises two important questions. First, how does one harmoniously combine users' individual adaptations, community-of-users design, and usage-informed professional design? Second, if/when useful, how does one allow users to interoperate software more generally than designed-for-adaptation systems? The rationale for this latter question is that users use specific systems (e.g., scenario editors), generic systems (e.g., office tools), and online resources. Attempting to enhance the ones designed for them with adaptations means goes in the right direction. Taking a broader perspective, however, users should be empowered to adapt and/or interoperate all the artifacts they use, as necessary given the instruments they develop. This is a hot topic. In web development different technical solutions make it possible to create mash-ups by combining and aggregating different resources. How to generalize this to more traditional software remains an open question, and may require a change of design focus (see Section 6.1).

6. DISCUSSION

In this section we successively present and discuss connections with other design perspectives, AT-related HCI works, and other theoretical approaches.

6.1. Similarities To and Differences From Other Design Perspectives

Requirements Analysis, UCD, and Metadesign

The perspective we propose is not contradictory to requirements analysis and UCD. However, it leads to specific stances and suggests reinterpreting these approaches and adapting them.

Based on the importance of the notion of functional value, the perspective we propose suggests that requirements analysis and UCD must acknowledge that users may develop legitimate unexpected usages. Artifacts must allow for a wide range of usages. Considering users' needs should therefore not lead to an implicit or explicit objective of making sure that users develop *the correct* interpretation and *the intended* usage. Moreover, by focusing on instruments rather than on artifacts, the perspective we propose suggests the importance of considering the developmental aspects of usage and their psychological, social, and work practice dimensions. Rather than focusing on how users interact with software, we suggest focusing on how users interact

with the tasks they consider and, in this context, use software. Moreover, fundamentally, because humans evolve, instruments and systems of instruments are dynamic. Requirements analysis and UCD must therefore be framed into a cyclic life cycle such as the one presented in Figure 3, and it must be put as a premise that offering adaptation features is necessary.

Taking this perspective, metadesign and EUD are techniques for implementing this life cycle and offering users different ways of adapting software to their needs at both macro- and microlevels. The framework we propose and the metadesign SER general model match: proposing a seed (designing initial artifacts and their a priori adaption features) and then conducting periodic reseeding (offering new artifacts and/or new task-related features and/or new adaption features). In our framework, however, this seed/reseed cycle and the underlying required analyses are anchored in an instrumental perspective, and the focus is on software appropriation and related adaptations. Similarly, work related to ecologies of artifacts must be reinterpreted in terms of instruments and systems of instruments, acknowledging that the artifact–instrument relation is not a 1:1 relation and that it is dynamic. Moreover, here again, ecology of artifacts phenomena must be considered at both macro- and microlevels (e.g., adoption of new artifacts and artifact appropriation and adaptation processes, respectively).

Designing Activity Spaces

Kaptelinin and Bannon (2012, p. 294) proposed changing the focus of design from systems (i.e., products) to Technology-Enhanced Activity Spaces (TEAS), which they define as "spatially and temporally organized configuration of resources, including digital technologies, which enable an individual or a group to carry out one activity or several coordinated activities." The rationale is to allow users conduct "intrinsic practice transformations," initiated and accomplished by them when contradictions within existing practices appear, and implemented by assembling artifacts "from the outside" via integrative technologies such as metatools or connectors.

The theoretical perspective and the analyses at the basis of this proposal are coherent with the ones we developed. In particular, Kaptelinin and Bannon (2012, p. 290) emphasized that "designers cannot simply impose on users their understanding of how people in the setting should act and how they should use technology—even if the understanding is based on a thorough analysis of users' needs and requirements." The notion of intrinsic practice transformation aptly characterizes the user-adaptation and community-of-users design perspectives outlined in this article. However, these authors push the idea that UCD has intrinsic limitations to a greater extent than we do. Where we consider identifying adaptation features relevant for users' activities and users' development as a key question, they consider metadesign and EUD as useful but intrinsically insufficient, suggesting that the focus must be on allowing users to interoperate software artifacts more generally than designed-for-adaptation systems, and propose to solve this issue by designing activity spaces.

The TEAS notion is inspiring and consistent with our perspective. Nevertheless, in our view, user–task and user–system relationship analyses and "embedded" means of adaptation remain important for many reasons. First, if nothing else, interconnecting artifacts that have not been prepared for raises open technical issues. Second, shifting the tasks of reflecting on activities and organizing pieces of available code to users may correspond to different realities. For instance, if one considers users such as "teenagers" (i.e., generally and acontextually), the list of artifacts and interactive systems they use may be very unstable, and studying the instruments they develop may prove difficult. One may also hypothesize that such users will welcome being given the means to creatively interoperate their artifacts themselves. In direct contrast, most workers (e.g., teachers) have analyzable work practices, operate within analyzable work settings, use systems designed for them, face a considerable workload, and are first and foremost concerned with achieving their professional tasks. For such users and contexts, studying the uses of systems and offering means of adaptation makes sense and goes in the right direction. Moreover, it paves the way for users to gradually consider as a given that artifacts should be adaptable and interoperable, which may create social pressure for designers to work in the direction of the ideas suggested by the TEAS approach.

6.2. Discussion With Respect to AT-Related HCI

The way we build on the instrumental genesis theory to explain appropriation advances the understanding of how this theory, which is not specifically related to software artifacts, can be related to software characteristics and design, on both software engineering and technical issues. This leads to consider new theoretical notions (e.g., alignment and instrumentalization episodes) and research questions. Links with recent work related to ecologies of artifacts are also clarified. It must be noticed that, because we are interested in software adaptation, we focused on instrumentalization. However, instrumentation is the foundation upon which instrumentalization takes place, and these dual dimensions must be considered together. Considering how to facilitate instrumentalization processes makes sense only if users consider that the system provides features that are useful for achieving the task they consider.

With respect to AT-related works, the analytical work presented in this article is consistent with that of Bødker and Klokmose (Klokmose, 2011), even though the latter study addresses a more general issue than ours (design issues in general). As mentioned, the human–artifact model they propose may be used to frame usage analyses following the perspective we proposed.

Combining AT/instrumental genesis perspectives and the ecology of artifacts perspective raises the question of whether these perspectives are compatible. The answer, arguably, is affirmative: Considering ecologies of artifacts does not require a change of theoretical lens. Fundamentally, these activity-centered perspectives focus on the user as an active actor, not on a given artifact and its use. AT acknowledges that human activity is multimediated (Bødker & Klokmose, 2011). Kaptelinin and

Bannon (2012) also argued that, although some aspects may need further development, AT concepts may be used to consider multiple artifacts and activities. Bødker and Klokmose (2012) showed how the AT-based human–artifact model could be used to address artifact ecologies. The instrumental genesis theory, which is rooted in the analysis of work practices, natively considers that users use a variety of artifacts (Rabardel, 2001, 2003).

Another classical notion related to appropriation is the affordance notion, which is a subject of ongoing debate in the HCI community. The perspective we developed is consistent with an AT-oriented and mediated-action perspective on technology affordances (Bærentsen & Trettvik, 2002; Kaptelinin & Nardi, 2012). (These authors, however, developed slightly different perspectives.) Following Bærentsen and Trettvik, we may say that the system's adaptation features are not affordances as such: They become affordances if the user relates them to his or her activity. Creating conditions for this to happen is thus an important dimension of designing for appropriation.

As raised by Bødker and Klokmose (2011), AT-oriented HCI is often criticized for being too complex and abstract. We see the work presented in this article as an example linking theoretical, design, and implementation considerations. Theory is used not to predict what will happen (how users will appropriate a system) but to provide an understanding of phenomena in question. This helps disentangle fundamental mechanisms (e.g., the constructive nature of activity) and contingent aspects, and helps one consider appropriation issues in a much more holistic way than focusing on artifacts. Identifying principles and guidelines from general theoretical bases, however, requires clarifying the theories with respect to the issue in question (e.g., dissociating different perspectives on design continued in use). Considering design also raises new theoretical issues (see Section 5.4). Finally, the clarifications developed for design considerations contribute to some of the general questions raised by the original theories, for example, relating user design and designer design.

6.3. Discussion With Respect to Other Theoretical Approaches

Genres and Organizational Theory

The genre perspective is different from our approach but proposes similar ideas, leads to similar conclusions, and proposes interesting methodological tools (and, actually, has some links with AT). Works such as Spinuzzi and Zachry (2000) or Yates and Orlikowski (2002) show how genre systems structure collaborative work as tacit habitual mechanisms and as devices. Another interesting aspect of the genre approach is that it acknowledges idiosyncratic, divergent understandings and uses of artifacts as they develop within a given cultural-historical milieu (Spinuzzi, 2001). With respect to links with AT, Engeström (2009, p. 308) referred to genres as "classifications of artifacts-plus-intentions" and "ways of seeing what acts are available and appropriate in a given situation." He argued that genre and activity may be seen as complementary units of analysis, although the anchoring of the genre notion in written works may be a limitation when considering multiple modalities.

The theoretical model framing Mackay's analyses (the "structurational model") is borrowed from organizational theory (Mackay, 1990). It addresses the different types of influences between institutional properties, technology, and human actors. This perspective suggests different factors that may influence users' decisions to customize software, such as institutional properties (e.g., available resources, standards or norms), external events (e.g., an office change), constraints and opportunities, and individual factors (e.g., previous experiences, technical skills, or expectations relative to a job or the technology). Here again, the approaches are different but not incompatible.

Semiotic Engineering

A very different theoretical perspective that also sheds some light on why/how designers' and users' views may diverge is semiotic engineering (de Souza, 2005). Taking a semiotic perspective, software interfaces are signs referring to, and meaning, what designers had in mind. Interfaces may thus be seen as encoding a message that designers send to users, whose content is the designers' understanding of users' activities (e.g., tasks, ways of acting). As designers' and users' meaning-making mechanisms may differ, and the message is one-shot, users may interpret the message (may associate meanings) in an unexpected way.

Although theoretical bases differ, this semiotic perspective shares an important idea with the developmental perspective we presented. According to semiotic theories, meaning is an ongoing process, which stabilizes only temporarily; users constantly revise their meanings or generate others ("unlimited semiosis process"). As a consequence, the challenge is not only that predicting users' interpretation is difficult (actually, impossible) but that these interpretations also evolve. This is consistent with the dynamicity of instruments and systems of instruments we highlighted. Another similarity is that, in keeping with this dynamic perspective, semiotic engineering advocates EUD and may be used to characterize tailoring actions, for example, according to whether they effect meanings encoded in the system (de Souza & Barbosa, 2006). Taking this perspective, usages related to unexpected functional values (e.g., managing empty scenarios as a way to model from patterns) may be seen as "repurposing" the system (to use de Souza's and Barbosa's words). The way (de Souza & Barbosa, 2006, p. 419). see such usages is coherent with our perspective:

> Although it may not be considered EUD in the sense that the system has not been modified, it is certainly an expansion of the scope of system usage, especially in view of the user's unlimited semiosis process, where new meanings associated to (the software features) become part of (the system) possibilities.

The original focus of semiotic engineering is to help designers communicate their message to users as a way for users to correctly interpret that message. Actually, making the designers' message more explicit is useful both when users' considered tasks and ways of acting correspond to designers' views and when they diverge, as it helps users understand the divergence and deal with it. Studying how the semiotic perspective may

be associated with ours to help users adopt a reflective activity on their use of artifacts and engage in adaptations is an interesting perspective.

The importance of considering multiple, coexisting interpretations of artifacts has also been raised in other studies. As an example, Sengers and Gaver (2006) argued that divergent interpretations of systems may not be a problem and that promoting a single correct interpretation may not always be necessary. Actually, we find the idea of "correct" interpretation debatable.

Ecology of Artifacts-Related Works

In the approach we developed, we propose considering both the detailed characteristics of individual systems and ecologies of artifacts. This diverges from much prior work.

An underlying implicit perspective of much prior work that considers ecologies of artifacts is that, because users adopt new artifacts if/when the current ones no longer align with their needs, the individual characteristics of these artifacts, including their adaptation features, are not really important. Here again, as mentioned when analyzing links with the TEAS approach, we disagree.

First, the fact that users may skip from one artifact to another is only one reality (e.g., teenagers' use of interactive technologies). In many cases, however, some, if not all, of the artifacts used by users are explicitly or implicitly imposed (e.g., by an institution, company, and/or community), and many users use artifacts that they would either change or not use if given the choice.

More fundamentally, with respect to appropriation phenomena, we see the macrolevel of the dynamicity of ecologies of artifacts and the microlevel of the appropriation and adaptations of artifacts as intrinsically related. As an example, let us consider studying how/why users such as teachers, while attempting to improve their work by sharing resources, get confused by an overwhelming number of e-mails generated by this new practice and decide to use new artifacts such as wikis. This may be analyzed at the macrolevel in terms of adoption or replacement of artifacts. While using the "old" artifacts, however, these actors developed appropriation processes and instruments that may play a role in why and how they adopted the new artifacts. Appropriation and development processes related to an artifact, and how they are supported or constrained, contribute to the way users act within their ecology of artifacts, and how this ecology functions.

Other Works

A general approach to appropriation in the light of how technical, social, political, and economic factors influence each other is proposed in Carroll, Howard, Vetere, Peck, and Murphy (2001). DeSanctis and Poole (1994) also focused on social structures. On the psychological side, some works attempted to address the appropriation issue from a cognitive perspective. As an example, Salovaara (2008) suggested viewing appropriation as a process of perceiving resources facilitated by previous and

immediate experiences, which may be modeled as usage schemata (this comes close to the interpretative dimension of users' adaptations mentioned in Section 4.3). In terms of general perspectives, a comparison of a dialectical focus, such as the one introduced by instrumental genesis, versus a causal focus, such as the one developed by the task–artifact cycle (Carroll & Rosson, 1992), is developed in Bødker and Klokmose (2011). The general idea according to which software should be adaptable may also be found in works by Shirky (2004), who introduced the term "situated software" to refer to "personalized, localized software that has evolved organically and has been created by the community that uses it," or Balasubramaniam, Lewis, Simanta, and Smith (2008), who suggested considering "opportunistic software" by combining several software systems via a scripting language to satisfy needs not anticipated in the design of the individual system.

7. CONCLUSIONS

We developed a theoretical perspective on designing for appropriation that considers the productive and constructive dimensions of activity, the individual and collective aspects of design continued in use, and the fact that users act within ecologies of artifacts. We proposed a set of statements summarizing the theoretical considerations, some general implications for design, some methodological and technological means, and some research questions raised by this perspective. We also reviewed various other perspectives on software appropriation and adaptation.

We see these elements as contributing to a theoretical understanding of software appropriation phenomena, to a definition of "designing for appropriation" as a well-defined and addressable issue rather than as an oxymoron, to offering designers means to conceptually and technically address appropriation issues, and finally to the theoretical reflection on design methods.

The implications for design highlighted by this work are coherent with guidelines that may be found in the literature, for example, Dix (2007), Robinson (1993), Pekkola (2003), Blackwell et al. (2001), or Fischer et al. (2004). Reconceptualizing the issue and developing a theoretical account, however, presents both theoretical and practical added values. Building from an understanding of the phenomena at stake offers a conceptual basis for framing requirements analysis and design processes, a rationale for existing design guidelines, and a basis for shaping these guidelines. It also identifies new areas of study that may help advance theoretical understanding and practical means.

In our view, designing for appropriation should be considered a primary goal of any design method. We have seen that, although possible perspectives present significant differences, they may correlate with and/or complement each other. Therefore, we believe that when considering a given setting or software, designers should consider what different theoretical perspectives (e.g., developmental, genre, or semiotic perspectives) say about appropriation and the implications for design, if any. Appropriation is a complex process, and complex processes require analyses from multiple perspectives.

NOTES

Acknowledgments. The author thanks Pierre Rabardel for his help in clarifying some of the theoretical aspects presented in this article.

HCI Editorial Record. First received June 19, 2014. Revision received October 24, 2014. Accepted by Terry Winograd. Final manuscript received May 31, 2016.

REFERENCES

Bærentsen, K. B., & Trettvik, J. (2002). An activity theory approach to affordance. *Proceedings of the NordiCHI 2002 Conference on Human–Computer Interaction*. New York, NY: ACM.

Balasubramaniam, S., Lewis, G. A., Simanta, S., & Smith, D. B. (2008). Situated software: Concepts, motivation, technology, and the future. *IEEE Software*, *25*(6), 50–55. doi:10.1109/MS.2008.159

Béguin, P. (2003). Design as a mutual learning process between users and designers. *Interacting with Computers*, *15*(5), 709–730. doi:10.1016/S0953-5438(03)00060-2

Béguin, P. (2007). In search of a unit of analysis for designing instruments. *Artifact*, *1*(1), 12–16. doi:10.1080/17493460600610830

Bertelsen, O., & Bødker, S. (2002). Interaction through clusters of artifacts. *Proceedings of the ECCE 2002 Conference on Cognitive Ergonomics*.

Blackwell, A., Britton, C., Cox, A., Green, T., Gurr, C., & Kadoda, G. (2001). Cognitive dimensions of notations: Design tools for cognitive technology. *Cognitive Technology: Instruments of Mind*, 325–341.

Bødker, S., & Klokmose, C. N. (2011). The human–artifact model: An activity theoretical approach to artifact ecologies. *Human–Computer Interaction*, *26*(4), 315–371. doi:10.1080/07370024.2011.626709

Bødker, S., & Klokmose, C. N. (2012). Dynamics in artifact ecologies. *Proceedings of the NordiCHI 2012 Conference on Human-Computer Interaction*. Copenhagen, Denmark: ACM.

Carroll, J., Howard, S., Vetere, F., Peck, J., & Murphy, J. (2001). Identity, power and fragmentation in cyberspace: Technology appropriation by young people. In G. Finnie, D. Cecez-Kecmanovic and B. Lo (eds), *Proceedings of the 12th Australasian Conference on Information Systems (ACIS 2001)*. Vol. 1, 95–102.

Carroll, J. M., & Rosson, M. (1992). Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Transactions of Information Systems*, *10*, 181–212. doi:10.1145/146802.146834

Cerratto Pargman, T. (2003). Collaborating with writing tools: An instrumental perspective on the problem of computer support for collaborative activities. *Interacting with Computers*, *15*(6), 737–757. doi:10.1016/j.intcom.2003.09.003

Clot, Y., & Faïta, D. (2000). Genres and styles in work analysis: Concepts and methods. *Travailler*, *4*, 7–42.

Costabile, M. F., Mussio, P., Parasiliti Provenza, L., & Piccinno, A. (2008). End users as unwitting software developers. *Proceedings of the ICSE 2008 International Workshop on End-User Software Engineering*. New York, NY: ACM.

de Souza, C. S. (2005). Semiotic engineering: Bringing designers and users together at interaction time. *Interacting with Computers*, *17*(3), 317–341. doi:10.1016/j.intcom.2005.01.007

de Souza, C. S., & Barbosa, S. (2006). A semiotic framing for end-user development. In H. Lieberman, F. Paternò, & V. Wulf (Eds.), *End user development* (Vol. 9, pp. 401–426). New York, NY: Springer.

DeSanctis, G., & Poole, M. S. (1994). Capturing the complexity of advance technology use: Adaptive structuration theory. *Organization Science*, *5*, 121–147. doi:10.1287/orsc.5.2.121

Dix, A. (2007). Designing for appropriation. *Proceedings of the British HCI Group Conference*, *2*, 28–30.

Dourish, P. (2003). The appropriation of interactive technologies: Some lessons from placeless documents. *Computer Supported Cooperative Work*, *12*, 465–490. doi:10.1023/A:1026149119426

Engeström, Y. (2009). The future of activity theory: A rough draft. In A. Sannino, H. Daniels, & K. D. Gutiérrez (Eds.), *Learning and expanding with activity theory* (pp. 303–328). New York, NY: Cambridge University Press.

Fischer, G. (2003). Meta-design: Beyond user-centered and participatory design. *Proceedings of the HCI 2003 International Conference on Human–Computer Interaction*. Boca Raton, FL: CRC Press.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A. G., & Mehandjiev, N. (2004). Meta-design: A manifesto for end-user development. *Communications of the ACM*, *47*(9), 33–37. doi:10.1145/1015864

Folcher, V. (2003). Appropriating artifacts as instruments: When design-for-use meets design-in-use. *Interacting with Computers*, *15*(5), 647–663. doi:10.1016/S0953-5438(03)00057-2

Gantt, M., & Nardi, B. A. (1992). Gardeners and gurus: Patterns of cooperation among CAD users. *Proceedings of the CHI 1992 Conference on Human Factors in Computer Systems*. New York, NY: ACM.

Gueudet, G., & Trouche, L. (2009). Towards new documentation systems for mathematics teachers?. *Educational Studies in Mathematics*, *71*(3), 199–218. doi:10.1007/s10649-008-9159-8

Gueudet, G., & Trouche, L. (2011). Communities, documents and professional geneses: Interrelated stories. In G. Gueudet, B. Pepin, & L. Trouche (Eds.), *From text to "lived resources": Curriculum material and mathematics teacher development* (pp. 305–322). New York, NY: Springer.

Jung, H., Stolterman, E., Ryan, W., Thompson, T., & Siegel, M. (2008). Toward a framework for ecologies of artifacts: How are digital artifacts interconnected within a personal life? *Proceedings of NordiCHI 2008 Conference on Human–Computer Interaction*. New York, NY: ACM.

Kaptelinin, V. (2003). Learning with artefacts: Integrating technologies into activities. *Interacting with Computers*, *15*(6), 831–836. doi:10.1016/j.intcom.2003.09.006

Kaptelinin, V., & Bannon, L. (2012). Interaction design beyond the product: Creating technology-enhanced activity spaces. *Human-Computer Interaction*, *27*(3), 277–309.

Kaptelinin, V., & Nardi, B. (2006). *Acting with technology: Activity theory and interaction design*. Cambridge, MA: MIT Press.

Kaptelinin, V., & Nardi, B. (2012). Affordances in HCI: Toward a mediated action perspective. *Proceedings of the CHI 2012 Conference on Human Factors in Computer Systems*. New York, NY: ACM.

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., . . . Wiedenbeck, S. (2011). The state of the art in end-user software engineering. *ACM Computing Surveys*, *43*, 3. doi:10.1145/1922649.1922658

Kuutti, K., & Syrjänen, A. L. (2011). Beyond user studies and appropriation: Developmental work research. *Proceedings of the CHI 2011 Conference Workshop on Appropriation and Creative Use: Linking User Studies and Design*. New York, NY: ACM.

Law, E., Roto, V., Hassenzahl, M., Vermeeren, A., & Kort, J. (2009). Understanding, scoping and defining user experience: A survey approach. *Proceedings of the CHI 2009 Conference on Human Factors in Computer Systems*. New York, NY: ACM.

Lieberman, H., Paterno, F., Kalnn, M., & Wulf, V. (2006). End user development: An emerging paradigm. In H. Lieberman, F. Paterno, & V. Wulf (Eds.), *End user development* (pp. 1–8). The Netherlands: Springer.

Mackay, W. E. (1990). *Users and customizable software: A co-adaptive phenomenon* (Unpublished doctoral dissertation). Massachusetts Institute of Technology, Cambridge.

MacLean, A., Carter, K., Lovstrand, L., & Moran, T. (1990). User-tailorable systems: Pressing the issues with buttons. *Proceedings of the CHI 1990 Conference on Human Factors in Computer Systems*. New York, NY: ACM.

Mørch, A. I. (1997). Three Levels of End-User Tailoring: Customization, Integration, and Extension. In M. Kyng & L. Mathiassen (Eds.), *Computers and design in context* (pp. 51–76). Cambridge, MA: MIT Press.

Mørch, A. I. (2011). Evolutionary application development: Tools to make tools and boundary crossing. In H. Isomäki & S. Pekkola (Eds.), *Reframing humans in information systems development* (pp. 151–171). New York, NY: Springer.

Pekkola, S. (2003). Designed for unanticipated use: Common artefacts as design principle for CSCW applications. *Proceedings of the SIGGROUP 2003 International Conference on Supporting Group Work*. New York, NY: ACM.

Quinones, P. A., Teasley, S. D., & Lonn, S. (2013). Appropriation by unanticipated users: Looking beyond design intent and expected use. *Proceedings of the CSCW 2013 Conference on Computer Supported Cooperative Work*. New York, NY: ACM.

Rabardel, P. (2001). Instrument mediated activity in situations. In A. Blandford, J. Vanderdonckt, & P. Gray (Eds.), *People and computers XV— Interactions without frontiers* (pp. 17–30). London, UK: Springer-Verlag.

Rabardel, P. (2003). From artefact to instrument. *Interacting with Computers*, *15*(5), 641–645. doi:10.1016/S0953-5438(03)00056-0

Rabardel, P., & Bourmaud, G. (2003). From computer to instrument system: A developmental perspective. *Interacting with Computers*, *15*(5), 665–691. doi:10.1016/S0953-5438(03)00058-4

Rabardel, P., & Waern, Y. (2003). From computer artefact to instrument for mediated activity [Special issue]. *Interacting with Computers*, *15*(5&6).

Robinson, M. (1993). Design for unanticipated use. *Proceedings of the CSCW 1993 European Conference on Computer Supported Cooperative Work*. New York, NY: ACM.

Russell, D. R. (2009). Uses of activity theory in written communication research. In A. Sannino, H. Daniels, & K. D. Gutiérrez (Eds.), *Learning and expanding with activity theory* (pp. 40–52). New York, NY: Cambridge University Press.

Ryan, W., Stolterman, E., Jung, H., Siegel, M., Thompson, T., & Hazlewood, W. R. (2009). Device ecology mapper: A tool for studying users' ecosystems of interactive artifacts. *Proceedings of the CHI 2009 Conference on Human Factors in Computer Systems*. New York, NY: ACM.

Salovaara, A. (2008). Inventing new uses for tools: A cognitive foundation for studies on appropriation. *Human Technology*, *4*(2), 209–228. doi:10.17011/ht/urn.201506212397

Sengers, P., & Gaver, B. (2006). Staying open to interpretation: Engaging multiple meanings in design and evaluation. *Proceedings of the DIS 2006 Conference on Designing Interactive Systems*. New York, NY: ACM.

Shirky, C. (2004). *Situated software*. Retrieved from http://www.shirky.com/writings/situated_software.html

Sobreira, P., & Tchounikine, P. (2012). A model for flexibly editing CSCL scripts. *International Journal of Computer-Supported Collaborative Learning*, *7*(4), 567–592. doi:10.1007/s11412-012-9157-9

Sobreira, P., & Tchounikine, P. (2015). Table-based representations can be used to offer easy-to-use, flexible, and adaptable learning scenario editors reference. *Computers & Education*, *80*, 15–27. doi:10.1016/j.compedu.2014.08.002

Spinuzzi, C. (2001). Software development as mediated activity: Applying three analytical frameworks for studying compound mediation. *Proceedings of the 2001 SIGDOC International Conference on Design of Communication*. New York, NY: ACM.

Spinuzzi, C. (2002). Modeling genre ecologies. *Proceedings of the SIGDOC 2002 International Conference on Computer Documentation*. New York, NY: ACM.

Spinuzzi, C., & Zachry, M. (2000). Genre ecologies: An open-system approach to understanding and constructing documentation. *Journal of Computer Documentation*, *24*(3), 169–181. doi:10.1145/344599.344646

Stevens, G., Quaisser, G., & Klann, M. (2006). Breaking it up: An industrial case study of component-based tailorable software design. In H. Liebermann, F. Paterno, & V. Wulf (Eds.), *End user development* (pp. 269–294). New York, NY: Springer.

Stolterman, E. (2008). The nature of design practice and implications for interaction design research. *International Journal of Design*, *2*, 55–65.

Tchounikine, P. (2011). *Computer science and educational software design—A resource for multidisciplinary work in technology enhanced learning*. New York, NY: Springer.

Trigg, R. H., & Bødker, S. (1994). From implementation to design: Tailoring and the emergence of systematization in CSCW. *Proceedings of the CSCW 1994 Conference on Computer Supported Cooperative Work*. New York, NY: ACM.

Trouche, L. (2003). From artefact to instrument: Mathematics teaching mediated by symbolic calculators. *Interacting with Computers*, *15*(6), 783–800. doi:10.1016/j.intcom.2003.09.004

Trouche, L. (2004). Managing complexity of human/machine interactions in computerized learning environments: Guiding student's command process through instrumental orchestrations. *International Journal of Computers for Mathematical Learning*, *9*(3), 281–307. doi:10.1007/s10758-004-3468-5

Vergnaud, G. (1998). Towards a cognitive theory of practice. In J. Kilpatrick & A. Sierpinska (Eds.), *Mathematics education as a research domain: A search for identity* (pp. 227–240). Kluwer Academic.

Wulf, V., Pipek, V., & Won, M. (2008). Component-based tailorability: Enabling highly flexible software applications. *International Journal of Human-Computer Studies*, *66*(1), 1–22. doi:10.1016/j.ijhcs.2007.08.007

Yates, J., & Orlikowski, W. (2002). Genre systems: Structuring interaction through communicative norms. *Journal of Business Communication*, *39*(1), 13–35. doi:10.1177/002194360203900102

Ye, Y., & Fischer, G. (2007). Designing for participation in socio-technical software systems. *Proceedings of the UAHCI 2007 International Conference on Universal Access in Human-Computer Interaction*. New York, NY: Springer.