Chapter 11
Building Socially Embedded Technologies: Implications About Design

Federico Cabitza and Carla Simone

11.1 Motivations and Background(s)

It is something of an open secret that every now and then resonates with a tinge of disgruntled resignation in the specialist literature of the last 30 years or so and even more recently (e.g., Lyytinen and Robey 1999; Klein and Jiang 2001; Shapiro 2005; Pan et al. 2008; Warkentin et al. 2009): approximately half to two-thirds (if not more, in critical domains like healthcare; see Heeks 2006) of information systems (IS) projects fail. This fact strikes one even more in light of the almost universal recognition that the practice of information systems development has undergone a radical transformation in this period and has abandoned naive strictly structured life cycle methods of development in favor of more flexible, dynamic, and multidisciplinary approaches.[1]

Indeed, with the developing complexity of information systems, the tighter coupling of their components, and the increasing opacity of internal functioning, there is little wonder that a purposively contrarian theory like the "normal accident theory" by Perrow (1999) with respect to computer-supported organizations (Szewczak and Snodgrass 2002, p. 64) and their infrequent but potentially harmful technologically driven failures (see, e.g., Rochlin 1998; Ash et al. 2004) has continued to provoke until recently (Weick 2004) by stating: "failures are normal."

[1]If this is true, one could argue that it is probably also because some principles and sensibilities typical within the HCI, CSCW, and PD fields have so to say "trickled down" in the "consciousness" of IT practitioners in the "real" world (cf., e.g., Shapiro 2005; Fitzpatrick and Ellingsen 2012).

F. Cabitza (✉) • C. Simone
Dipartimento di Informatica, Sistemistica e Comunicazione, Universitá degli Studi di Milano-Bicocca, Milan, Italy
e-mail: federico.cabitza@disco.unimib.it

V. Wulf et al. (eds.), *Designing Socially Embedded Technologies in the Real-World*, Computer Supported Cooperative Work, DOI 10.1007/978-1-4471-6720-4_11

A known essayist has even quite provocatively argued around the conjecture that digitizing (or informating) organizations and their work is of little or no use for their competitiveness and performance or even worse has a potential to corrode existing competitive advantages, for instance, by homogenizing complex business processes (Carr 2004).

Of course, there is little comfort in being aware, especially in the EUSSET community, that computing-related failures seem largely due to organizational and social rather than technical factors (Pan et al. 2008; Kaplan and Harris-Salamone 2009). Yet, even framing what "success" really is, how to detect it and gauge the extent a project is successful, can be seen as primarily a social and cultural effort rather than a merely technical one (Wilson and Howcroft 2002; Thomas and Fernández 2008): different approaches can focus more either on quantitative and economic indicators (e.g., DeLone and McLean 1992; Cooke-Davies 2002) or, at the opposite extreme, on users' perception and satisfaction (e.g., Myers 1995; Goodhue and Thompson 1995).[2]

Irrespective of our peculiar inclination to consider typical information systems as "good" or "bad" when such a computer-based system fails or, on a microscale, exhibits a relevant failure, two possible conjectures are likely to emerge, related to two opposite perspectives to the issue: what we denote as the Daedalus conjecture and the Icarus conjecture from the famous myth of the first manned flying machine. The former one is the attitude of who tends to speculate on users that misinterpreted or misused the system, assuming that the machine's design is proper and fit for intended use assuming correct operating procedures ("feather wings were not supposed to be used too close to the sun"). Conversely, the latter one is the attitude of one who fingers poor design and claims a right to pursue objectives also beyond the idea of "intended use," which he/she considers to be usually shortsighted and to limit real use excessively ("who the hell would employ mere wax to stick a pair of wings together?"). Whatever the cause anyway, Icarus comes to a bad end.

In this chapter, we argue in favor of a third approach toward technology's shortcomings, which we will articulate in what follows to scrape it off of its outward and possibly misleading nuances of provocativeness and end-unto-itself oddity: the perspective according to which computing tools cannot be really defined "a priori" and "in vitro" by someone external to their use, i.e., the "mighty designer," but only be iteratively constructed in the wet "mangle of practice" (Pickering 1995) by end users themselves. We propose this perspective to try to go beyond both the typical illuminist optimism of the "mighty designers" and the fatalistic attitudes of technological Cassandras in tackling the so-called software development crisis: a general condition that probably regards computer technology development since its beginnings that has been explicitly debated since the end of the sixties and said to

[2]This spectrum of utility evaluation seems to oscillate between the different stances of the philosophers who tried first to understand how to gauge usefulness and satisfaction, Jeremy Bentham and John Stuart Mill, respectively.

have in those years primed the so-called "software engineering" field (Haigh 2010) and its rational design-centered methods and methodologies to achieve IT success.

We like to characterize this approach in terms of a "contrarian and alternative mythology" with respect to the mainstream, design-centered mythology that has been dominant since the dawn of software engineering. We speak of mythology after Harris and Henderson (1999), who make the point that "while our hardware technology has improved by orders of magnitude, and our software has grown comparably more complex, the relationship between people (individually or in groups) and computers has only improved incrementally. In some cases, it has even *deteriorated*" (p. 88, our emphasis). The authors address the reason why it is so difficult "to translate [research insights] into comparable improvements in the usability (and more generally, the social integration) of computers" by advocating the adoption of a "better mythology for system design" in alternative to the standard mythology. This latter encompasses a set of "myths"[3] summarized in what follows:

- The parts of the system must interact according to a preestablished harmony defined during its design.
- The job of a designer is to discover, clarify, and when necessary invent the rules that define that harmony and then embed them into the computer system.
- The users must interact with the system in terms of the language or ontology that these rules create.

This mythology sustains the legitimacy of a process that is carried out by experts (in IT design) with the participation of experts (in their own practices) in order to represent and direct the unfolding of the production of computer-based information systems in an orderly manner in the face of chaos (AA 2001). The main merit of Harris and Henderson (1999) is to have shed light once again on some taken-for-granted assumptions. This is also our aim. In fact, only when "the limited and inaccurate perspective on work and technology imposed by the standard myths of both organization and system design [have been recognized], we can start to search for more effective approaches and write better myths around them" (ibid).

Although we also agree with the tenets Harris and Henderson (1999) proposed within their "mythology for the long term" almost 15 years ago,[4] in our little alter-

[3]Here and in the following, the word myth is not opposed to any truth fact, but it is rather used as synonym of "archetypical story" to indicate one possible stance, among many other ones as much as legitimate and reasonable. On the other hand, we keep using the term mythology for its powerful and evocative connotation, although probably the most indicated term would be "metanarrative," in the sense after Lyotard (1986), i.e., set of narratives that emphasize particular aspects of the practice of IT development and that, in doing so, do not drive practice in any strong sense but rather tell it, legitimate it, and "shape it by helping each participant construct and frame their account of their practice" (Harris and Henderson 1999, p. 89).

[4]Although the interested reader can refer to the original paper, we here summarize the main high-level recommendations contained in the mythology proposed by Harris and Henderson (1999): that we should i) honor every particularity, even those that do not fit the regularities imposed by the organizational rules; ii) honor accommodation, i.e., the "ad hoc elaboration of rules in use"; and iii) honor change, which is an intrinsic and unavoidable feature of a real world system.

native mythology, we will go a step further by arguing around the idea that the very conception of design that we are all well used to (and many of us also are fond of) should be challenged and indeed conjectured to be one of the most decisive factors leading to manifest failure.[5] A similar argument was put forward by Bryant (2000).

Our conjecture, while similar, limits itself to submitting that the main assumption underlying the modern idea of design, i.e., that it is proper and safe to distinguish between design and use, is too disconnected from practice, although it heavily relies on representations of the latter, and it is nourished as the central element of a reductionist framework where the process of system development is more or less rationally phased down into subcomponents that are ontologically distinct and where responsibilities are assigned on the basis of nominal competencies somehow reified in terms of specialized roles.

We are not saying a rational and engineering approach is bad per se, but we submit that it reflects conceptualizations of "what complex is" and "how to cope with complexity" (Cabitza and Simone 2012c) that are based on widespread misunderstandings of the complexity theory (Paley and Eva 2011; Maguire and McKelvey 1999)[6] and hence can bring unsubstantiated expectations and convictions.[7]

Thus, however radical this point may seem, we will take it seriously in order to justify the argument that such a conception (and the related professional activity) is not really necessary to build any successful computational, material artifact with which users have to interact to have their work done. We propose an alternative approach that manages without formal or conceptual design by which, as discussed in (Cabitza 2011), we contest the necessity and primacy of such kind of design in the development of computational interactive applications and information systems that are to be embedded in social cooperative settings. With "design," we here denote that specific phase of the larger development process in which professional analysts meet some (or many) user representatives and/or their managers to draw more or less formal models of how work is and should be accomplished (the "flow of work") and produce detailed specifications of the needs of the various stakeholders involved and of how the computational system will support work to fulfill needs and expectations. Thus, although we take the term to encompass business analysis, requirement elicitation, conceptual modeling, process (re)design, specification formalization,

[5]We will certainly not try to prove this conjecture, as we could never get over the causality fallacy that such a proof would entail (i.e., post hoc, propter hoc).

[6]For instance, Paley (2007) makes the point that many researches that declare a focus in complex systems do actually refer to the open systems thinking, which between the 1960s and the 1980s was aimed at replacing the Tayloristic organization-as-machine metaphor with the metaphor of organization-as-organism; more curtly, Maguire and McKelvey (1999) assert that most of the references to the complexity theory in the IT-oriented literature are not dissimilar from "mere retellings of old tales, [which use] complexity terminology tacked on retrospectively, gratuitously, and, in many cases, quite awkwardly."

[7]Moreover, Ivan Illich was among the first thinkers to denote a similar phenomenon as "principle of (paradoxically) counterproductivity": once most practices are institutionalized and engineered, they backfire on some of the stakeholders (Illich 1977).

and analysis, in what follows, we will use the general term "design" for brevity's sake. Other authors have cautioned against the fictional and ritualistic nature of this activity (e.g., Robey and Markus 1984; Robinson and Bannon 1991; Nandhakumar and Avison 1999). We contest the myths in which this ritual is considered necessary (see also, e.g., Shipman and Marshall 1999) and substantially unquestionable (see also Blackwell and Green 2008; Cabitza 2014a). Conversely, we will argue in favor of alternative myths according to which all the layers pertaining to human-computer interaction in the broadest sense – the model, the control, and the view – can be realized by the composition of elementary components without any "rational" design input and be put to work by end users alone, eventually (but not necessarily) flanked by IT professionals that are explicitly called to play the role of catalysts of a "reaction" that pertains to the dynamics of complex (socio-technical) systems in situ (Cabitza et al. 2014a, b).

We are aware that also our argumentation encompasses some myths, the most notable of which is that the "end user that can develop her own artifacts" somehow. Moreover, this alternative "mythology" is not "new" or "original" in any strong sense. It does however resonate with complementary recent discourses, as we shall see. We present these ideas again as a contribution of the foundation of an alternative way to build socially embedded systems.

The rest of chapter will be articulated as follows: in Sects. 11.2 and 11.3, we will briefly gather suggestions from two distinct discourses on which to ground our different mythologies: performativity thinking (see Sect. 11.2) will help us reappraise the value for IT development of the subterranean river that connects many influential thinkers from Nietzsche to Suchman and will provide us the conceptual space to think of system development differently. The metaphor of the *bricoleur* (see Sect. 11.3) will suggest to us a new strategy for building computer-based support in the wild. This pathway will lead us toward an alternative proposal in Sect. 11.4 discussed further in Sects. 11.5 and 11.6. Section 11.7 will look at a research agenda coherent with this alternative mythology for IT system development.

11.2 The Rediscovery of Performativity

Many things difficult to design prove easy to performance. (Samuel Johnson 1759)

In this section, we will first consider what we mean when we advocate that the alternative mythology we are envisioning should produce a "performative turn" in IT system development. Then, we will consider how the performative discourse has already come into design-related mythologies, in order to highlight the specific strand we aim to renovate with our proposal.

11.2.1 The Performative Turn

The expression “performative turn” is usually used to indicate two related aspects that we nevertheless prefer to distinguish for clarity’s sake. On the one hand, it indicates a historically circumscribed research program, which has received an increasing interest in the last 15 years by researchers involved in cultural and social studies, like the science and technology studies field (notably Pickering and Latour) and related disciplines like ethnology, anthropology, sociology, and linguistics; in this former case, the term “turn” indicates the aim of this research endeavor to investigate an alternative way to look at how people interact, work, and share knowledge in social settings with respect to more mainstream strands like the pragmatic and realist paradigms, endorsing the claim that people create and recreate meaning and knowledge in social settings through performance (Van House 2009) and that even social reality itself is “created” while people “do things.” As Law and Singleton (2003) put it:

The differences between realism and pragmatism are important, but neither share the performative assumption that reality is brought into being in the process of knowing. Or, to put it more precisely, neither would assume that the object that is known and the subject that does the knowing are co-produced in the same performance, or that the epistemological problem (what is true) and the ontological question (what is) are both resolved (or not) in the same moment.

The performative approach shared the critique of systemic, fully specified, and rationally conceived abstractions (e.g., with the nonrepresentational theorists[8]) and drew on the metaphor[9] of “performance” to reflect “a growing discontent with the traditional social sciences and their understanding of practices as texts or representations of genuinely symbolic concepts,” to express “the reversion from systems of representations to processes of practice and performance,” and to focus on “the active social construction of reality rather than its representation” (Dirksmeier and Helbrecht 2008).

In an attempt to summarize the recent, and quite protean, discourse about performativity in two lines, after Bramming et al (2012), we highlight three intertwined aspects: (i) reality is understood as incessant creation or practice; (ii) matter itself is understood as “entangled intra-relation”; and, of course, (iii) individuals do not preexist their interactions in any essentialist, objectivistic sense.

There is a second connotation of the expression “performative turn” that we now want to refer to. This latter, rather than a specific research program, can be better characterized as a sort of “sensitivity to specificities of materially heterogeneous

[8]It should be noted though that “a performative perspective does not delete the idea of representation, but rather views it as a specific aspect of performativity” (Jensen 2005), in that it focuses on the activity of representing, planning, and modeling rather than on the material outcome of those practices.

[9]Here and elsewhere, we use the term “metaphor“ in the Nietzschean sense, as something that is used to impose order and intelligibility on a world that we cannot access directly.

events with special reference to differences and relations between performances" (Jensen 2002). This sensitivity has followed in the last century or so a peculiar karstic trend: it has recurred a number of times by different authors of different cultural milieus, and, although each time it was capable to gain a strong interest, this was never sufficient to establish itself as the mainstream thought in any of those milieus and somehow submerged until a next thinker contributed in its reappraisal.

In this sense, therefore, the idea of a "performative turn" evokes a more historical attitude, which was exhibited by individuals that have deliberatively turned away their focus from the allures of representationalism to embrace a more action-oriented and embodied perspective. The term "turn" thus indicates the will to reverse the ontological premises that the world is populated with particular objects, entities, and configurations that exist in and of themselves and that are endowed with particular essential qualities (Jensen 2002, p. 67) to consider objects, "not singular entities, but rather textures of partially coherent and partially co-ordinated performances" existing through multiple situated practices.

This sensitivity, or will, or discontent with representational/conceptual tenets indicates a sort of "fil rouge" that binds together thinkers like Nietzsche, Heidegger, Derrida, Pickering, and Latour[10] and some relevant feminist theorists (Bath 2009; Butler 1993) like Judith Butler, Karen Barad, and, especially for her involvement in the IT debate, Lucy Suchman (just to mention a few of those authors that influenced our understanding of the performative approach). A common trait among these thinkers seems then to be the need to find a viable alternative to representationalist tenets (i.e., stances that could be called as Cartesian or simply "modern"[11] (cf., e.g., Rorty 1991)), to shift the focus from questions of correspondence between models/representations and reality to matters of practices/doings/actions (Barad 2003).

In short, a performative approach asks us as observers of social settings to abandon the idea that these are sets of "object that are," to embrace the idea that they are made of "events that do." In doing so, it gives us a "resource to counter the positivist stance which essentializes categories and naturalizes the qualities of the entities whose stable existence it posits" (e.g., gender as a fixed attribute of a person) (Licoppe 2010). The concept of performativity therefore invites us to abandon the Kantian notion of "thing per se" (at least in system design) to recognize the relational and manifold nature of any perceived phenomenon, irrespective of its

[10]The fil rouge binds together unsuspected associates, like Pickering and Latour. One thing that unites these thinkers, for example, is that they are both "happy enough" to speak of material agency in nature without imputing any intentionality to the word "agency" (Pickering 1995, p. 6).

[11]Yet, we agree with Jensen (2002) when he points out that "the performative turn is a way to refuse the choice between the modern and the post-modern. The modern is about order purity. The post modern is a celebration of fragments and disorder. The performative turn is a series of claims and sensitivities that try to reach a fractional space in between. Something that is beyond the mono-dimensionality of modernity and beyond the free-floating multi-dimensionality of the post-modern. In this sense it has much in common with the parts of the Actor Network Theory tradition that claim to be non-modern."

seeming solidity,[12] as well as the co-constitutive entanglement of the social and the technological (i.e., material) and "the performance of the emergent sociomaterial assemblage" (Orlikowski 2007). According to this perspective, "meaning" is thus seen as an emergent phenomenon (or an epiphenomenon) of interaction (Hug 2010) but, even beyond this point, as a transient aspect of embodied interaction (Dourish 2001) that cannot be really decoupled from situated action (Suchman 2006) nor caught in abstract terms.

In this vein, researchers adopting a performative turn put first in their research agenda the study of the contingencies of time, space, technology, materiality, or discourse, "the heterogeneous sociomateriality and real-time contingency of performance," as Suchman (2006) calls them (p. xii): all things that the more classical "representational" model of thinking that is typical of "twentieth century technoscience" (Suchman 2004), i.e., the one assuming a detached observer that studies real objects and their essential properties in an objective world (or that designs and puts new objects into the world), escapes either consciously or unaware with profound consequences also on the conception of the role of technology in society and of its "designers" (Orlikowski 2007).

11.2.2 The Performativity Fil Rouge

In order to frame how the concept of performativity can influence IT system design in practical ways, we have to briefly outline the fil rouge mentioned above, which binds together influential thinkers of the last 150 years with the foundations of the CSCW approach to system design. To this aim, we have first to make a clear distinction between the discourse on performativity we are interested in and the so-called performance studies. These latter are usually at stake where scholars and researchers in the IT literature use expressions like "designing for performativity" (Morrison et al. 2010), "the role of performance in design research" (Jacucci et al. 2005), or "performing design." These expressions are more related to the traditional meaning of performance (Dirksmeier and Helbrecht 2008), as "showing of a doing" (cf. Grimes) or "activity before a particular set of observers" (cf. Goffman),[13] and they point all more or less to the "artistic" side of the discourse on performativity and as such they tend to "preserve," if not enhance, the creative role of designers instead of contributing in the overturn of the necessity of the idea of design.

[12]To support the legitimacy of the performative turn, we here recall that our ancestors (i.e., Latin, Greek, and Old English) used the words "res," "pragma," and "thing" (respectively) in order to denote an affair, a deed, a business, or an assembly (Telier 2011, p. 1), as well as the matters that were discussed and deliberated in such occasions and meetings. In other words, subject and object did not need to be disentangled on such occasions.

[13]It is nevertheless worthy of note that the meaning of performance as "performing a play" or "playing a drama" is much later than the more general meaning of "carrying out a promise" or "carrying in effect something" that dates from the sixteenth century.

Conversely, the concept of performativity we refer to is rooted in the Nietzsche's seminally deconstructive analysis of the relation between words and the world and in his powerful intuition according to which looking for a specific "doer" behind any action is recognized as an arbitrary and unnecessary (and indeed confounding) act.[14] This seminal contribution was then taken up by phenomenologist scholars, notably Heidegger, who further articulated the idea that the only way of being of human (i.e., Dasein) is engagement in practices (Existenz) (Riemer and Johnston 2012), that these latter depend on equipment[15] for their performance, and that the relationship between this latter and Dasein is fundamentally co-constitutive (Turner 2005). Many affinities can be then found between Heidegger and J. L. Austin (see, e.g., those discussed in Glendinning 1998), who introduced the concept of performative utterance to account for the capacity of human speech to act, i.e., have an effect in the material world, rather than just simply describe reality in terms of "true" and "false" statements; that notwithstanding, Law questioned the orderly taxonomy proposed by Austin and claimed that "all statements are in the slippery space between performative and constative," thus turning "the question of constative vs. performative [. . .] into an empirical question, and thus potentially an object for a sociology of performances" (Jensen 2002).

Years later, approximately at the same time as these concepts were taken up in the IT design arena by Winograd and Flores (1986) in their reappraisal of Austin's (and Searle's) elaboration of the so-called speech acts, the performative "fil rouge" unfolded again in the works of Andrew Pickering. We are referring to those contributions where this author made a clear distinction between a "representational idiom" and a "performative idiom" in scientific and technology-oriented discourses (Pickering 1995) and in particular for our design-related discourse, when Pickering (2008) contrasts the modern technoscientific approach to the design of things with the approach followed by British cyberneticians, like Beer, Ashby, and Pask, i.e., a hands-on experimental, performative, and non-representational one. At the same time, other authors drew upon the critical reinterpretation by Derrida (Simon 2010) of Austin's original differentiation between performatives and constatives, most notably Judith Butler and Karen Barad. These latter elaborated a complex concept of (posthumanist) performativity around the repetitive, or citational, aspects of performance, i.e., its ability to produce materiality. In this view, social structures, like rules and categories (such as gender), are not preexistent attributes of a given object or its behavior, but rather they are continuously produced through processes of repetition and social legitimization. This conviction echoes, but also in some way goes beyond, the views animated by Wittgensteinian philosophy that recognizes

[14]We are referring to the famous passage in *The Genealogy of Morals* where Nietzsche pointed out that "there is no 'being' behind doing, acting, becoming: 'the doer' is merely a fiction added to the 'doing'. Doing is all" (original: es giebt kein 'Sein' hinter dem Thun, Wirken, Werden; 'der Thaeter' ist zum Thun bloss hinzugedichtet, — das Thun ist Alles).

[15]Equipment can be seen as a term which denotes those things, or artifacts, that the Dasein encounters in fluent use, entangled and experienced in performance, when they are ready- to- hand (Zuhandenheit).

how, due to the intrinsic underspecification of human behaviors (Schmidt 2011a, b), it is the practice that determines the rule rather than the opposite and that invites us to abandon an "objectified and detached view of rules and procedures as external objects with fixed properties, to a performative view where rule following is characterized as a typically emergent, distributed and artifact-mediated activity" (D'Adderio 2008).

11.2.3 Performativity for IT System Development

All that said, one could rightly wonder what the performative turn, as it has been characterized above, has to do with the discourse regarding IT design in socio-technical settings and, above all, if there is anything new. We are aware that some of the performativity tenets like paying attention to "the negotiations between actors" (Wagner et al. 2010, p. 67) and the question of when design stops and use begins (cf., e.g., Brand 1995) "may seem old to people within the CSCW tradition" and related ones (i.e., HCI, PD, and the like).[16] That notwithstanding, we believe that this perspective can be fruitful along both the practical and conceptual dimension.

11.2.3.1 On the Practical Side: Toward New Meaningful Development Cycle*s*

From the practical point of view, only a few contributions so far refer to the performative tenets explicitly with respect to design; for instance, Jensen (2008) advocates a reorientation of both the understanding and (less clearly) the practice of the process of IT design (or more specifically of CSCW design) as performativity; to this aim, he submits recommendations to keep in mind performative aspects in the design process, such as that "neither humans nor technologies determine each other" and that "materiality might trick us in practice." Unfortunately, the author falls short of clarifying how a performativity-aware disposition or "relativizing one's own ontology" (although certainly a useful exercise) could also "revitalize design" and really change the practice of IT design. With a more practical attitude, Danholt (2005) makes an argument about the performative nature of prototypes, by suggesting that prototypes "affect users in concrete, material, bodily ways in situ."

Recognizing the performative nature of prototyping is then related to recognizing that this way of designing artifacts is "mutually transformative for users as well as for the technology, a process of co-construction of humans and artifact"; if design

[16]This was honestly admitted by Jensen (2008), who has nevertheless advocated a better consideration of these ideas within those traditions. However, two years later, Bratteteig et al. (2010, p. 31) have conversely recognized that "the performative turn in post-structuralism is perhaps under-articulated in design research."

"is considered to be performative," it is recognized as "an emergent process where the end result is not predicated by either users or designers, but [is] an outcome of the process. [. . .] Performativity thus also means that the existing is continuously performed and reiterated in order to persist, which means that the existing is also always under construction and transformation. Slight changes in the way things are done lead to novel existences. Performativity thus implies a continuous possibility of transforming the existing."

While we would fully subscribe to these conclusions, we notice how user-centered, and even participatory design, approaches (let alone any approach within the more traditional, engineering mythology), in which users are considered to hold important knowledge on their practice and, *in virtue of this competence*, are involved in the design process (in some form), are nevertheless still considered to be end users and their practices as preexisting the design process and somehow invariant to the task, *in its essential traits*. Thus, while prototyping and participatory prototyping, especially when prototypes are not merely representational ones (i.e., mock-ups) but rather are working gears (like in the framework presented in Harel 2008), can make the distance between design and use (and hence designers and users) shorter, the co-construction of these prototypes usually takes place in a controlled and delimited environment ("in vitro" rather than "in vivo"). In so doing, the performative dimension of the development process is still kept at the margin of the real and never-ending (and very aptly depicted as loop-closed) process of the task-artifact cycle (Carroll et al. 1991), where both the task and the artifact coevolve as a whole and at a different pace.

Within a performative strand, such a cycle would likely resemble a more intertwined figure, where the task cannot be considered without the artifact with which it is accomplished and the artifact alone is just inert accoutrement outside the task. Taking seriously that "the social organization of work does not pre-exist in any precise or detailed way, but is constituted 'in the [artifact-mediated] doing' by practitioners" (Buescher et al. 2001) suggests then that tasks occur only when artifacts are used and artifacts make sense to practitioners only when these are put to work. In other words, there is no dualistic thing but situated action, which emerges from the indissoluble entanglement of tasks and artifacts, like in a variation of the widely known Taijitu symbol. It goes without saying that entanglements cannot really be designed, as "the take-up, modification and rejection of technology in a work setting, and the [conseguent] accommodation of work practices that take place around a developing technology, are radically unknowable and unpredictable" (Buescher et al. 2001) till they actually occur.

11.2.3.2 On the Conceptual Side: Back to the Future

The conceptual contribution is no less important if we accept what Schmidt (1999) once pointed out, i.e., that "Lucy Suchman's radical critique of cognitive science and the 'situated action' perspective she proposed has played a significant role in defining the CSCW agenda and has become a shared frame of reference to many, perhaps most, of us." Since the publication of *Plans and Situated Action* (1987), the discourse around the concept of "situation" has become more prominent in system design and underlies the main tenets of the EUSSET's "Situated Computing Manifesto."[17] This focus on "situation," rather than on performativity, has resulted, we would suggest, in a certain ambiguity, perhaps due to its apparent roots in the concept of a (static) place (cf. Latin situatio, site).[18] As pointed out by Clancey (1997, p. 23), "the overwhelming use of the term situated [. . .] since the 1980s has reduced its meaning from something conceptual in form and social in content to merely 'interactive' or 'located in some time and place'." Suchman (2006) herself admitted that the passage where she had written that "the situation of action can be defined as the full range of resources that the actor has available to convey significance of his or her own actions and to interpret the actions of others" could be erroneously "taken to imply that 'the situation' exists somehow in advance of action and that it could at least in principle be fully enumerated and represented in the form of a model to be referenced" and therefore as something that can be drawn by some professional (i.e., the designer) before actually going "where the action is" (Dourish 2001). Conversely, "the sense of the situation [Suchman is] after is a *radically performative and interactional one*, such that action's situation is in significant respects constituted through, or stands in a reflexive relationship with, ongoing activity" (p. 125, our emphasis).

This remark cannot be underestimated. Indeed, when Suchman exposed the main themes pertaining to her decades-long research in the field of HCI in the preface of *Human-Machine Reconfigurations* (a reprint of *Plans and Situated Actions* that was enriched by new footnotes and additional chapters), she mentions: "the irreducibility of lived practice, embodied and enacted; the value of empirical investigation over categorical debate; the displacement of reason from a position of supremacy to one among many ways of knowing in acting; the heterogeneous socio-materiality and real-time contingency of performance; and the new agencies and accountabilities effected through reconfigured relations of human and machine" (Suchman 2006, p. xii). It is for us indicative that Suchman did not mention "situated action" nor situatedness. Here we briefly recall that the former concept was originally chosen "to underscore the view that every course of action[19] depends in essential ways on its material and social circumstances" (p. 70) and the latter term was not originally used by Suchman, although hundreds of scholarly papers associate it with her

[17]URL: http://www.thinkinnovation.org/eusset-has-just-engineered-the-manifesto-of-situated-computing/ (accessed 03-Sept-2014). Archived at WebCite on 03-Sept-2014 [http://www.webcitation.org/6SJclKI3B]

[18]This could have also laid the concept of situatedness open to some representationalist drifts: cf., e.g., the connotations acquired by the term "context," among which that of "container-like" (Suchman 2006, p. 19), in IT-related discourses about "context-aware systems."

[19]Including planning itself or "calling out a plan as a self-standing artifact": cf., respectively, p. 17 and 21

work[20] and has been the object of some criticisms,[21] among which we recall here the point by Ciborra (2006) regarding the paradoxical and somehow extraordinary lack in such concept of any affective, human, but we would also say performative, element.[22] In the same vein, also the current interests on either "situated software" (Balasubramaniam et al. 2008) or "situated computing" can be questioned. As Suchman put it:

> I believe that the argument made [in 1987] holds equally well today, across the many developments that have occurred since. The turn to so-called situated computing *notwithstanding*, the basic problems identified previously – briefly, the ways in which prescriptive representations presuppose contingent forms of action that they cannot fully specify, and the implications of that for the design of intelligent, interactive interfaces – continue to haunt contemporary projects in the design of the "smart" machine. (Suchman 2006, p. 3, our emphasis)

Thus, the IT system design discourse periodically contains terms and expressions that have the potential to overturn the traditional oppositions between abstraction vs. materiality, representation vs. performance, and between different kinds of design, e.g., the one that "solidifies and stabilizes procedures and classifications" (Orlikowski 1992a) and the one that "continues in use" (Carroll 2004): these terms and expressions nevertheless end up by getting like "muted," although they still remain as "sensitizing concepts [:::] which draw attention to important features of work and provide guidelines directing research in specific settings" (Crabtree et al. 2001). It is as if those "sensitizing concepts" were always put into a sort of seventh room of Bluebeard's castle, where they are seemingly kept alive, honored, and dolled up but actually in a state of harmless captivity, with no real influence on actual practices and on the inner convictions of the practitioners involved in design. This could be just the plain consequence of an "engineering education [which] had over-invested in analytical technique and scientific understanding at the expense of the practical, 'hands-on', the creative, the reflective, the social, the constructive, the ethical, the economic" (Bucciarelli 2003, p. 295).

In conclusion, we assert the topicality of the performative turn (especially in the sense of the intellectual legacy argued above) and advocate the concept of performativity to be taken more seriously in the future for at least two reasons: first it refers to a "doing" explicitly and in that it differs from the keywords like "situation," "situated(ness)," and "context" which all refer to a "state of being." Hopefully this could be enough to avoid falling victim to the Scylla

[20]In *Human-Machine Reconfigurations*, Suchman speaks of situatedness only once and only to challenge the meaning intended for such term by Rodney Brooks, the MIT engineer that questioned symbolic representational approaches in the field of robotics, as she found such meaning "evacuated of sociality."

[21]Including people, like Lave and Wenger (1991), who lament the vagueness of the definition itself of situatedness

[22]Ciborra (2006) writes: "'Situated' is the translation of the German 'befindlich'; situatedness is 'befindlichkeit'. [The former term] not only refers to the circumstances one finds himself or herself in, but also to his or her 'inner situation', disposition, mood, affectedness and emotion."

of essentialism/representationalism (Maturana and Varela 1992) and facilitate the reappropriation of Suchman's lesson, at least within the CSCW community.

Second, we believe that the performative view, in its nature anti-conceptual, anti-representational, and against the divide between design and use (i.e., practice), has a potential to bring us to the other side of the river (cf. the life-raft model mentioned by Buescher et al. 2001)) and let us assert that technology in practice (Orlikowski 2000) cannot really be "designed" but rather allowed to "emerge"[23] (Cabitza 2014). This would mark the shift with no regrets "from a focus on invention [we would say of design, Ed.], understood as a singular event, to an interest in ongoing practices of assembly, demonstration and performance. The shift from an analysis in terms of form and function to a performative account" (Suchman et al. 2002, p. 165). We re-propose this resolution within our alternative mythology as a way to bridge the literature contributions mentioned above and the following discourse on bricolage.

11.3 From Models to "Bricolages"

I often try out little bits
wheresoever they might fit.
The sages call this bricolage,
the promiscuous prefer menage . . . [24]

The discourse on the performative nature of socio-technical systems suggests we should recognize that designing for interaction and action is overambitious for its irreducible distance from the actual performance of the task. This would seem to cast a gloomy light on any constructive stance about computer-based support of complex human tasks. However, what gives us "some hope" is that an approach, if not a method, can be taken toward the actual realization of technological scaffoldings (Orlikowski 2006) for collaborative complex socio-technical systems: bricolage.

In the context of IT design-related research, we draw heavily on the concept of bricolage for its "overall generative effect [which] seems to be more dependent on interaction rather than on some overriding design rationale" (Lanzara 1999, p. 347) and because "bricolage privileges combinatory logics, loose coupling, and garbage can processes" (ibid) and minimizes the prospect that any designed thing, no matter how well conceived, will necessarily fall short of avoiding the "law of unintended consequences" (Mansfield 2010).

Early authors to use the concept of bricolage in relation to design (in a wide sense) were (almost independently) Weick (1993) and Ciborra (1992). These studies, although largely, provided the conceptual background for many subsequent contributions that leveraged, or simply were inspired by, this metaphor. Among

[23]Of course someone has still to develop the technological artifact, and someone else pays the bills.

[24]Thomas Erickson, 2000, allegedly written upon reading a commentary for a special issue of CSCW Journal on Theory

these, we also consider the contribution by Buescher et al. (2001), one of the first to provide some concreteness to the notion of bricolage within the actual process of the development of computer-based information systems in organizational settings. In their work, Buescher et al. (2001) suggest a "'life-raft' model of systems development – a continuously unfolding bricolage of technologies to hand, requiring much patching and baling, with an unknown destination" (p. 17). In this "overarching framework within which newly developed technologies are set in place and helped to 'work'," they argue that the design process had to become more "immediate and continuous" in order "to cope with the deeply built-in uncertainty of the relationship between technical systems and work practices" (p. 22). They provide a concrete definition of bricolage in a CSCW context:

> Bricolage can be described as 'designing immediately', using ready-at-hand materials, combinations of already existing pieces of technology – hardware, software and facilities (e.g., Internet providers) – as well as additional, mostly 'off-the-shelf' ones. It therefore also involves design as assembly [and] requires investigation of the process of assemblage as well as designing for it. (p. 23)

We substantially agree with the points regarding the idea of "design as assembly" and the immediacy of the bricolage-oriented approach. Yet, we interpret immediacy in terms of "unmediated spontaneity" rather than in terms of "ad hoc quickness" and therefore bricolage as an activity mainly accomplished without the mediation of designers or IT specialists. At the same time, asserting that bricolage is "a description of the existing context," the general activity of bricoleur as well as its "(unforeseeable) outcome" (i.e., an assemblage of "things that work," the solution coming out from a particular round of development), and even a (presumably context-independent) "method for design" is rather catchall.

We therefore prefer the more focused definition proposed by Hartswood et al. (2000):

> Users need the opportunity that only their work can offer to explore fully the possibilities for adopting, and adapting to, new systems and artefacts. When this is allowed to happen, and given the right choice of technologies, development work can assume the characteristics of 'bricolage' – i.e., the rapid assembly and configuration of 'bits and pieces' of software and hardware – led by users acting within their own work settings, with IT specialists taking on the role of facilitator.

In this light, we propose to dissolve the usual distinction between a passive end user and a more active end user (the latter idea has been called a variety of things in the literature; see, e.g., Cabitza et al. 2014a, b) and hence to consider all users as (at least potential) bricoleurs, i.e., who in different circumstances can play either the role of who constructs and assembles the pieces of technology (whom we denote as "bricolant" bricoleur) or who exploits those assembled pieces by actively using them according to the situation at hand (i.e., a sort of "actant" bricoleur). This is the twofold meaning of the term "bricoleur" that we submit for the IT discourse. This can be clearly traced back to the specific archetype of bricoleur that Levi-Strauss (1966) introduced to contrast with the opposite archetype of "engineer." In our view, then, the latter can personify the rational designer that builds systems from scratch after, and in virtue of, a conceptual effort, while the former denotes the user that fabricates her own tools from available resources, being immersed in situated performances and contingencies. In his words:

> The bricoleur is adept at performing a large number of diverse tasks; but, in contrast to the engineer, he does not subordinate each one of them to the acquisition of raw materials and tools conceived and procured for the project: his universe of tools is closed, and the rule of his game is to always make do with 'what's available', that is, a set, finite at each instance, of tools and materials, heterogeneous to the extreme, because the composition of the set is not related to the current project, or, in any case, to any particular project, but is the contingent result of all the occasions that have occurred to renew or enrich the stock, or to maintain it with the remains of previous constructions or destructions. (Levi-Strauss 1966, p. 17)

For our purposes, the key motivations for focusing on the active roles of the end users can be found in three statements by Levi-Strauss (1966). Firstly, objects "are not known as a result of their usefulness; they are deemed to be useful or interesting because they are first of all known" (p. 9). This means that what is "useful" or not cannot be predetermined in terms of functional requirements, irrespectively of the competence of the analyst/designer, as these are necessarily decoupled from the actual availability of the corresponding functionalities in the workspace of users. Conversely, each work item is perceived by users to be useful if they have already internalized its function, that is, if they already know it and have made sense of it. This means that the bricoleur is someone that uses the objects she can find around her, but it is also necessary that their meaningful arrangement entails that he/she has previously been involved in some sense in the creation of those objects. Thus, bricolage is seen as an arrangement of predefined objects, where predefined here just means "defined before" and not "from above by someone else."

Secondly, a distinction between the engineer/designer and the bricoleur is made in virtue of "the inverse functions which they assign to events and structures as ends and means, [the designer] creating events (changing the world) by means of structures and the 'bricoleur' creating structures by means of events." (p. 22). This point is particularly important in view of how the performative stance sees every event.[25] This cautions us against regarding any structure that the designer could conceive as either enabling or constraining action as these structures may be changed in the process of their enactment, even if such a change is unintentional and unacknowledged (Orlikowski 1996). It also relates to the more manifest feature of the activity of bricolage: as said above, not only to make things out of the materials one has lying about but also to make sense of those materials according to an interpretive act that reinvents the objects (at least their meaning, their function, and their value) anew in the face of change and that is hardly anticipatable and mostly unplannable as it is also deeply conditioned by past interactions (we would also say "situated" of course).

[25]That is, as "an autonomic and contingent occurrence with its own conditions and its own time-structure, [in respect to which] the meaning of the past for the present is not fixed but radically ambiguous" (Dirksmeier and Helbrecht 2008), i.e., inextricably intertwined with the given situation

Thirdly and importantly, “the engineer works by means of concepts, and the bricoleur by means of signs” (p. 20). This is meaningful in light of the fact that “signs can be opposed to concepts [in that] whereas concepts aim to be wholly transparent with respect to reality, signs allow and even require the interposing and incorporation of a certain amount of human culture into reality” (p. 20). The idea of transparency here hints at a clear development recommendation: whereas the engineer aims to hide information[26] and to make his idea of, say, patient into a number of attributes unambiguously codified in a relational DBMS, underneath the application logic, the bricoleur instead needs to pay attention to what fields will represent the patient in his/her artifacts, arrange them the way she needs, fill them in on the basis of informal conventions and customs, as well as disregard them and create some new attribute/field at need, irrespective of any ideal model of that disembodied entity. Users and designers own distinct perspectives but nevertheless they have to interact to make the technological artifact fully operational in the target environment. Their collaboration has to be sought at a level that is different from artifact construction, as will be discussed in Sect. 11.5.3.

Moreover, this third passage also clearly requires: first, that a second but by no means less important activity of the bricoleur consists in a continuous and seamless accumulation of any sign that could help her make sense of the bricolage in practice; in so doing the bricoleur can enrich the bricolage artifact, i.e., its content as well and any kind of meta-content attached, like comments, tags, and nested threads of conversations that unfold around and about the tangible artifact. In short, bricolage is a continuous and creative “playing with signs.”[27] Second, this passage sheds light on the requirement that any computational support of the activity of the bricoleur must be oriented toward this continuous creative and interpretive activity, which, as we know (Berg 1999), can accumulate data as well as coordinate activities, toward the reconciliation of multiple, possibly diverging interpretations and above all toward the coexistence of these multiple and contextual incorporations, both in the local and in the global dimension.

This latter point is what makes us believe that the bricoleur-oriented mythology (as a specific kind of end user enabled by a specific kind of platform that we will outline in the next section) has the potential to oust the mythology oriented to the designer, i.e., the heroic and creative role that to some extent can be traced back up to the Renaissance imagination and that Hirschheim and Klein (1989) more prosaically denoted as the “systems expert.” This stereotype still distorts in professional practice (and not only there) the fragile symmetry of the Janus-like relationship between users and designers (Bowers 1991). In the next sections, we

[26]cf. the principle of encapsulation, which is defined by Grady Booch as “the process of compartmentalizing the elements of an abstraction”

[27]This passage is strongly influenced by the reading of Nietzsche by Derrida in “Structure, Sign, and Play,” where the Nietzschean perspective is related to “the joyous affirmation of the play of the world and of the innocence of becoming, the affirmation of a world of signs without fault, without truth, and without origin which is offered to an active interpretation.” Bricolage itself is a concept that urges us considering system development as a game-related social undertaking.

will speak about how a "laissez faire les bricoleurs" method can be flanked by a specific "logic of bricolage," in order to empower end users and have them become the builders of their own artifacts within their daily practices.

11.4 Toward Environments Supporting Bricoleurs

Everything that can be said, can be said clearly. (Ludwig Wittgenstein 1922)[28]

In this section, we would like to address how the discourse that we have outlined above can converge into a coherent and practical proposal for the development of interactive and collaborative information systems whose related mythology of system development should situate itself among the research lines that are emerging within the HCI field. As also recently pointed out by Ardito et al. (2012), these lines focus on concepts such as:

- Appropriation: i.e., the process by which technologies are understood and used by users in their own ways, possibly subverting the designers' intentions (Orlikowski 1992b; Dix 2007)
- Meta-design (Fischer and Giaccardi 2006): also denoted as "design for designers," a design paradigm which allows various stakeholders, including end users, to act as co-designers even at use time. Accordingly, software engineers do not design the final application, as in traditional design, but create software environments through which different stakeholders can contribute to the design of the final application
- End-user development (EUD) (Lieberman et al. 2006): a paradigm that focuses on the capability of systems to offer support at run time to empower users to develop their applications, blurring the distinction between design time and run time

As we will argue, the alternative proposal we advocate builds on but is distinct from these approaches. The term "appropriation" can be read as implying taking as one's own, a "thing" that has been constructed by someone else. For instance, Carroll (2004) writes of "the crucial role played by users' actions in completing the design process" and that "[technology appropriation] is actually part of the design process. The design of a technology innovation is completed by users as they appropriate it." We find then that the notion of appropriation is deeply ingrained in the design-oriented rhetoric.

In the same mold, meta-design is a term that explicitly refers to a phase of design, one programmatically aimed at investigating "techniques and processes for creating environments that allow 'owners of problems' (or end users) *to act as designers*" (our emphasis, Fischer et al. 2004). The main contribution that we want to retain from this framework is then the idea of "underdesign"; this notion relates to design for purposely "incomplete" systems that, once deployed, would allow for important

[28]*Tractatus Logico-Philosophicus*, 4.116

modifications by end users themselves, in the face of unexpected and unanticipated needs that show up at use time. Underdesign hints at a conceptual design that does not have the ambition to fully set the system up for its "embedment" in a complex socio-technical system, but it also hints at a design for the "underlayers," i.e., aimed at the construction of environments where applications can be developed with a strong interaction (co-design) between users and professional designers. Fischer et al. (2004) use the term "seed" to denote an underspecified application that users can complete during its use; the authors of this work also report about the action research initiatives that led to the construction of such environments by means of specialized editors (e.g., a map editor). The term seed is fully coherent with the idea that applications grow (Truex et al. 1999) and evolve with their environments but, in some way, this latter idea seems to clash with the claim that end users have to *act as designers*, if this means to envisioning how the application ought to be and ought to behave in the unknown future, even if this activity is performed by end users who play the "designer" role.

EUD is the approach which has most clearly and explicitly stated in its agenda (as well as in its name) the involvement of users in the construction of their technology and without expecting them to act as designers. This shows a strong affinity with the approach we discuss, especially if the meaning at stake for the term "development" is the original one mentioned in Sect. 11.1: the notion of a continuous and indefinite "unfolding" over time, pruned of its abstraction and differentiation from the actual work practices. This is the point that resonates more with the passage by Levi-Strauss reported in Sect. 11.3 where the end user, the *bricolant* bricoleur of our mythology, is expected to "work with signs instead of with concepts." Thus, *constructing* (or modifying) the artifact should not be seen as radically different from *working with* the artifact. The constructs and structures with which end users work should be familiar, like blocks and parts of the artifact itself, and conceived to be rearranged or created by composition from smaller subcomponents that are not ontologically different from their compounds (e.g., big field sections in forms are made of smaller fields groups, and these in their turn are but data fields).

Adopting a fully and coherent EUD approach has a strong impact on what kind of system is supposed to support the continuous bricolage-based construction of *convivial tools.*[29]

11.4.1 What Meta-system for End Users' Systems?

It is possible to distinguish between two main ways a system can act as a sort of meta-system for the development of an application by end users or at least for its tight adaptation to their needs. On the one hand, we can consider systems

[29]This expression is taken from Illich. A convivial tool is defined as "that which gives each person who uses it the greatest opportunity to enrich the environment with the fruits of his or her labour" (Illich 1973).

that primarily (or exclusively) support configuration. This regards the so-called "flexibility through control" of systems that offer ways for people to adjust settings and reprogram the system or otherwise technically adjust it (Dourish 1999). Yet, allowing the setting of more or less articulated parameters that affect the application's behavior or its appearance at the interface level entails little room for intervention by end users, since the set of elements is taken from a predefined (at design time) set of values and corresponding effects on the application at run time; accordingly, such systems allow for an involvement that is, in our view, too superficial (also literally speaking) and is constrained by some model of feasible action or by some feasible pattern in the "fitness landscape" (Mansfield 2010, p. 50) that results from precise configurations in the "design space."

On the other hand, other kinds of systems offer an environment that is "flexible through openness" (Dourish 1999), that is, a sort of "meta-system" by which users are supported in the creation of new systems and applications of different complexity, according to their needs and competences: macro-programming, visual programming, and programming by demonstration are among the solutions that are given to users to "encourage their participation in the design process" (Dourish 2001, p. 170). Here the risk may arise that the motivations and purposes of EUD-oriented researchers may clash with the scope and aims of the actual tools that are made available to the end users: specific features of the environment (or their absence) can introduce, or even impose, rigid models of practice and affect how end users build and maintain their equipment. This latter point relates to an important feature that environments enabling EUD practices should possess: we call this quality, *universatility*, to hint at something in between the traditional qualities of generality, universality, and versatility. While generality is usually defined as "the degree to which a software product can perform a wide range of functions" (Khosravi and Gueheneuc 2004) and hence serve multiple purposes, universality and versatility (from which *universatility*) regard the quality of being both general purpose and easily tailorable to the needs of specific settings and thus able to fit local needs. In other words, where generality refers to the typical quality exhibited by Swiss Army knives, that is, to have multiple specific functions to serve distinct but anticipated purposes, *universatility* refers to the quality of a tool that offers affordances that allow an open-ended set of usages (De Michelis 2003). Thus, a powerful environment has to be universatile enough to avoid imposing restrictions on the applications that it allows the construction of. Here the core of the problem lies in how this quality is guaranteed and on what conceptual premises are grounded.

11.4.1.1 Universatility Based on an Ontological Approach

The first way to make an environment general enough to be applied to any cooperative setting but also versatile enough to fit any (in principle) of its situated tasks is what we call the "ontological approach." This is an expression of the representational and objectivistic approach we discussed in Sect. 11.2: the designer of the environment decides how to guarantee wide customization on the basis of a pre-understanding of how actors behave in a number of recurring situations in multiple domains; consequently, on the basis of this understanding (which is based on deep introspection or more interactive and qualitative techniques), the designer conceives a set of "labels" that identify the "things" that users will handle, associates that classification scheme with intended universal building blocks, and provides users with those elements, all together with specific rules for their composition, so that they can (acknowledge and) make value out of that given model. A paradigmatic example of this approach was the Coordinator (Flores et al. 1988) 25 years ago: there the ontological claim was that actors coordinate their actions in terms of negotiation of commitments, and according to this model, the technology offered a universal set of possible categories to characterize setting-specific behaviors and routines. The assumptions underlying this technological proposal have been widely discussed, and contrasted, since then (e.g., Suchman 1994) but other examples of this approach still abound, both in daily life, for example, where reference management software force us to univocally associate our academic works or books with a specific category and, in recent academic research, for example, when users are called to categorize others' comments in public discussion with a system like Reflect (Kriplean et al. 2012).

In addition to systems where the ontological approach is adopted in an explicit form, we notice that such an approach can also act within an IT system *implicitly* (if not surreptitiously), especially in all those systems that adopt a characteristic or strong metaphor representing "the" one way in which humans allegedly organize their world: this is the case of the most famous (and nowadays notorious) "desktop metaphor," as well of some recent alternatives, like the metaphor of "story" proposed by De Michelis et al. (2009b); in both cases, users are called to associate the objects they work with a concept (i.e., the notion of file or of resource) and characterize it in terms of a category – being it the name of the folder in which the file is virtually stored (as well as the location of this latter in the "file system") or the name of a sequence of interactions with someone or about something (i.e., the topic of a conversation). The same phenomenon occurs in the ambit of context-aware or situated computing where, as mentioned in Sect. 11.2, tools to characterize a context or a situation are part and parcel of the design of the application itself, and they are based, again, on a predefined domain model that the users can only customize (or appropriate); this seems in basic contrast with the idea of context as "embodied action" that we share with Dourish (2004).

All these approaches, either explicitly or implicitly ontological, are grounded in the hypothesis that things could be described univocally or, at least, that the "name for a thing" would mean *that* thing irrespective of the setting where such a name is used and for what aim (cf., e.g., Mark et al. 2002; Anderson et al. 2008): this is the essence of an ontological stance. However useful this approach may be for ordering and retrieval purposes, any more or less structured "ontology" (in the broadest sense of a taxonomy, classification scheme, interaction metaphor, and the like) is conceived at *design time* and it is given to the users so that they make sense of their world in a way that can make some tasks more orderly efficient; yet, this approach may also hinder the support of other, possibly more "hidden" tasks: this mirrors platforms that provide users with functionalities that allow for some degree of tailorability but, as the latter is constrained within the boundaries of the metaphor itself, do not encompass functionalities to let the application (and its underlying ontology) evolve toward and align with the idiosyncratic customs of the users. The availability of such functionalities, and their subsequent use, could seriously undermine the consistency of the overall model and hence the effectiveness of the former tasks (e.g., Peters 2006).

11.4.1.2 Universatility Based on a Performative Approach

The main tenet of EUD has to do with giving users a more substantial role in technology conception, development, and evolution. Component design is proposed as an approach that allows users to tailor their applications by enriching them with suitable components offering specific functionalities (Mørch et al. 2004). This would require the application to be open to this type of tailorization (Stevens et al. 2006). Moreover, while "component thinking" seems natural for the integration of preexisting applications or in the assemblage of computational materials – for example, by using Lego Mindstorm environments (Rusk et al. 2008), we also have observed (Locatelli and Simone 2010) that in the construction of applications from scratch, this could be perceived as difficult by users, who usually see their application in a more holistic way than the component-based approach would suggest. A more radical stance is taken by Fischer and Giaccardi (2006) and their notion of meta-design. We have already expressed some reservations, at least on a purely conceptual level, regarding those platforms that would be aimed at making users act as "designers," rather than allowing them to construct their tools much like they already do with their traditional artifacts: i.e., by individual or bottom-up organized initiatives, trials and errors, progressive amendments, and patchwork or bricolage attitudes. Indeed, we have observed that actors in their everyday (working) life do not follow a traditional design-based approach to solve their problems and to construct the tools they need (Cabitza et al. 2013); this perception finds confirmation in a number of field studies (e.g., Carstensen et al. 1995; Morrison and Blackwell 2009; Blackwell and Morrison 2010; Handel and Poltrock 2011; Morrison et al. 2011) that focus on not-yet-digitized settings and that show a continuity in work practice development and paper-based tool construction that current technology is still not able to reproduce or guarantee.

Indeed, if we agree that one of the main issues here at stake is the gap between users and designers (in the traditional sense) – which is grounded in a conceptualization of design that will always prevent users from taking full control and responsibility[30] of the development process – we believe that this gap can be bridged only if design and the conceptual modeling activity that design implies are

[30]Beath and Orlikowski (1994) show how most of the user-centered development methodologies that put a strong emphasis on user involvement (they make the case of information engineering) actually relegate users to playing a relatively passive role during development and, in virtue of this, ask for a more clear responsibility for project outcomes. We stress here the need to give full

simply avoided and if the approach toward "technology co-construction" takes work practices "seriously," by avoiding any sort of compromise at the application level and by deriving the related consequences at the technological infrastructure level. In this vein, taking work practices seriously means to conceive technology construction as part of *work and articulation work*, in the same way as paper-based artifacts are constructed by actors when they need and use them (e.g., Morrison and Blackwell 2009) or, more generally yet, in the same way as users use work-arounds when their applications cannot be tailored in any satisfactory way (Cabitza and Simone 2013b).

We then argue that the sort of universatility that platforms must guarantee should be based on a radical performative approach for two reasons: first, according to an ontological approach, specificity and situatedness are reached by having actors apply a universal model locally and such a model can be both *adopted* and *adapted*, but adaptation is here only a sort of extension of its basic assumptions and first-instance concepts; conversely, a performative approach guarantees such locality by delegating the users to create *their own* essentially open, underspecified, incomplete, and even ambiguous "models," by which they can make sense of their do-it-yourself tools (Cabitza et al. 2013; Cabitza and Simone 2012c).

Second, adopting a performative approach "seriously" calls for the requirement of an environment that limits itself to providing primitives by which users can build their application in a *bottom-up fashion*, that is, in an emergent process of trial and error, and *while they work*, as a way to improve the odds that the application will really reflect and support their situated practices: if this construction were "extracted" from those practices and moved to a controlled environment of introspection, modeling, and ontological representation, we believe that we would again tap into a less than effective ritual, which is stuck with the conviction that the task-artifact entanglement can be really untangled without losing both (see Sect. 11.2). As Lanzara put it: "systems do not only operate or change in time, but are literally 'made with time'." Within a performative approach, as we discussed in Sect. 11.2, end users can be seen as bricoleurs who build their digital tools tapping into their tacit knowledge and their creative skills to build the portion of the IT artifact that comes closest to their work practices.

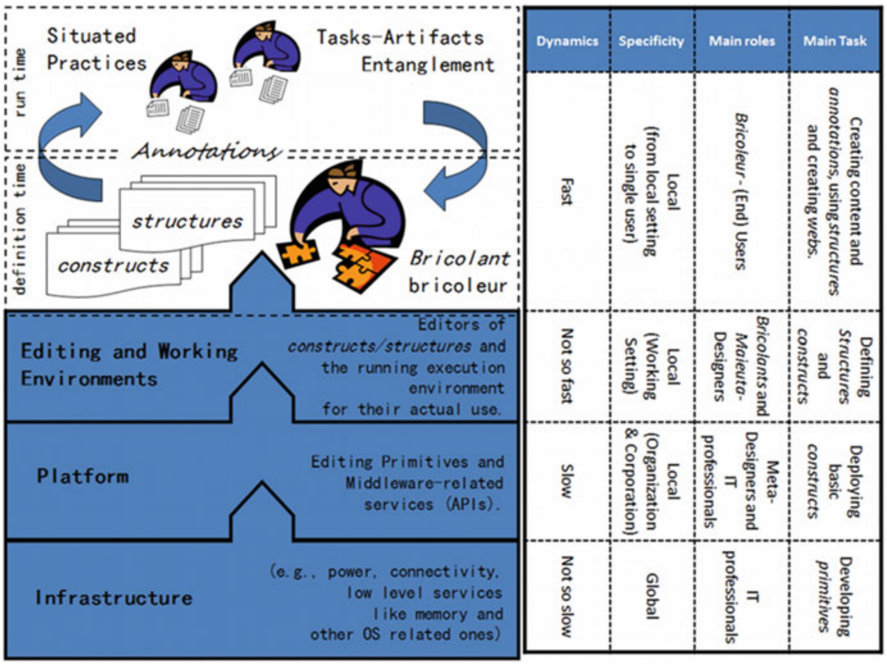
11.5 Concrete Steps Toward a Logic of Bricolage

In order to make a contribution toward the conceptual foundation of environments supporting the practice of bricolage in EUD terms, we will take inspiration from the point that Lanzara (1999) made on the importance of "transient constructs and persistent structures" (p. 332), which are seen as the results of "a practical, situated, context-sensitive mode of design that feeds on the dynamic tension between the requirements of change and stability." We also think that what he called the "logic of bricolage" emerges from the intertwined interplay of structures and constructs,

control, rather than only responsibility, to the community of users that will host the information system.

transiency and permanency, and universality and locality. This requires that an environment supporting bricolage does not provide users with sophisticated (i.e., semantically rich) modeling tools that facilitate the top-down construction of the application: from the conception of the "entities" involved, their attributes, their mutual relationships, and of the "business processes" where all these latter interact; rather this logic is supposed to offer to the users a set of "bricks" that they can arrange and compose together in a bottom-up fashion within a conceptually consistent environment (i.e., the rules of composition).

In order to envision such an environment, we propose a multilayered architecture that is partly inspired by the research accomplished in the COMIC project.[31] In this architecture, the layers that are closer to the greater source of uncertainty and unpredictability, that is, the layers that are closer to the users and their environment, are those which can be changed faster and to a greater extent. With reference to Fig. 11.1, we distinguish between an infrastructure, a platform, and environments for editing and working. The infrastructure is the set of available services that are


run time
definition time
Situated Practices
Tasks-Artifacts Entanglement
Annotations
structures
constructs
Bricolant
bricoleur
Editing and Working Environments
Editors of constructs/structures and the running execution environment for their actual use.
Platform
Editing Primitives and Middleware-related services (APIs).
Infrastructure
(e.g., power, connectivity, low level services like memory and other OS related ones)

Dynamics	Specificity	Main roles	Main Task
Fast	Local (from local setting to single user)	*Bricoleur* - (End) Users	Creating content and *annotations*, using *structures* and creating *webs*.
Not so fast	Local (Working Setting)	*Bricolants* and *Maieuta-* Designers	Defining *Structures* and *constructs*
Slow	Local (Organization & Corporation)	Meta-Designers and IT professionals	Deploying basic *constructs*
Not so slow	Global	IT professionals	Developing *primitives*

Fig. 11.1 A conceptual architecture for an environment supporting EUD bricolage (LOB keywords are in italics; each layer indicates its name and what it offers to the higher levels)

[31]The COmputer-based Mechanisms of Interaction in Cooperative work project was an EC-ESPRIT-funded basic research project No. 6225, from 1992 to 1995.

used by the computational platform that is specifically designed to support the bricoleurs in building and using their own tools.

This platform, in its turn, exposes specific services to make the bricolage-based information system possible and computationally augmented; with this aim, the platform instantiates a working environment where a persistent storage and a working memory, as well as an execution engine, are made available to the users, and it instantiates an EUD environment where users can create their building blocks and edit their tools; while working in this latter environment, users use specific visual editors to build both their constructs and their working structures that are put in operation when the working environment is "online."

The architecture outlined above sounds similar to the architectures usually proposed in literature to support modular approaches for application development, like the abovementioned component design or service orchestration (Papazoglou et al. 2007). In fact, although layers are actually common, what differs is their content. We aim at the definition of a platform implementing a general *generative* environment where bricoleurs can define, in principle, any sort of (collaborative) application, starting from elemental and universal building blocks and composing them according to universal composition rules. This idea is captured by the simplified "formal grammar" presented in Table 11.1. This is a first attempt to formalize the "logic of bricolage" toward its implementation.

Table 11.1 Generative productions of the logic of bricolage

<web-structure> ::= <layout-structure>+
<layout-structuree> ::= <topological-object>+
<topological-object> ::= <operand-construct> <coordinates >?
<operand-construct > ::= <constant> | <typed-variable> | <operator-construct>(<operand-construct >+)
<operator-construct> ::= <functional-operator-construct> | <actional-operator-construct>,
<annotation> ::= <style> <target-ref> + |<constant> <target-ref>+
< target-ref> ::= <functional-operator-construct> (<target>)
<constant> ::= <domain-values> | <multimedia-text>
<target> ::= <control-structure> | <topological-object> | <annotation>
<style> ::= <conventional-symbol> + | <operator-construct> (<target>)
<control-structure> ::= <rewriting-rule> | <connector>
<connector> ::= <functional-operator-construct> (<control-structure >+)
<rewriting-rule> ::= <condition> <action>+
<condition> : := <functional-operator-construct> (<state>)
<action> : := <operator-construct>(<state>)
<state> ::= <operand-construct>+

Legend: The LOB grammar is expressed in EBNF-like notation; therefore, the symbol "|" means "alternative," "<>" means "variable," "+" means "one or more occurrences," "_" means "zero or more occurrences," and "?" means "zero or one occurrences"; "domain values" are not specified and are the terminal symbols of the grammar (e.g., true and false).

In the following, we define the main elements of this grammar and illustrate them by means of the particular document-based information system environment that we have been developing in the last few years, called the web of active documents (WOAD, Cabitza and Simone 2010; Cabitza and Gesso 2011). The core concepts of WOAD can be summarized as follows: (i) the information system is parcellized in a set of hyperlinked active documents that can be annotated in all parts and sections and be associated with any other document, comment, and computational behavior; (ii) there is no rational and unified data model because users define their forms in a bottom-up manner and, in so doing, the platform instantiates the underlying flat data structures that are necessary to store the content that these forms will contain and to retrieve the full history of the process of filling in them; (iii) the presentation layer is in the full control of end users, who are called to both generate their own templates and specify how their appearance should change later in use under particular conditions (cf. the concept of affording mechanism proposed in Cabitza and Simone 2012b); and (iv) execution control is rule based. Users can define local rules that act on the documents' content and, as hinted above, change how documents look like (i.e., their physical affordances), to make themselves aware of pertinent conditions according to some cooperative convention or business rule like the need to revise the content of a form, or to consider it provisional, or to carefully consider some contextual condition (see, e.g., Cabitza et al (2009) for other examples of such conventions).

11.5.1 The Generative Environment

The grammar that supports the logic of bricolage (LOB) is based on the following first-class concepts or elements (see Table 11.1) where the technical terms *constructs* and *structures* are partly inspired by Lanzara's original contribution Lanzara (1999):

Constructing constructs. These are constructs that we denote as "constructing" because, first, they are *construct(ed)* during the inception phase of the platform within a cooperative setting, hopefully as result of a participatory design activity, and second because, once constructed, they are to be used as atomic "building blocks" by which the bricoleurs can create their working spaces and artifacts. We can further characterize these *constructing constructs*, distinguishing between *operand constructs* and *operator constructs*. Operators are all the feasible operations and micro-functions that users deem necessary to be performed over the operands; these latter are the most atomic data structures, components, and variables that the platform must make available in both the editing and working environments to be used in situated work practices. Both operands and operators are the "things" that are arranged and put together in the bricolage activity in order to, respectively, compose the artifacts and endow these of computational capabilities.

For example, in WOAD, the *operand constructs* are called *datoms* (document atom): these are any writable area with a unique name and a type (e.g., integer, string). A datom can recursively be a composition of one or more datoms: e.g., the “first name” datom (a string) and the “family name” one (also a string) can be combined into a “person name” datom that encompasses both. The *operator constructs* are a selection of atomic functions that include predicates and are denoted as *functional* in Table 11.1. For example, doctors from the medical setting described in Cabitza et al. (2009) required for their forms a construct to perform averaging and another one checking the occurrence of a value in a given set (i.e., the is-in construct), in addition to the standard arithmetic and Boolean operations. Operator constructs include also atomic functions that correspond to actions on specific operand constructs and are denoted as *actional* in Table 11.1. In the same vein as Simone and Schmidt (1993), we described a method to recognize and characterize these atomic components from the qualitative analysis of the paper-based artifacts used within a document-intensive work domain, i.e., two large hospitals (Cabitza 2011): in Table 11.2, we report the list of *actional operator constructs* that users agreed with us and they would need to apply to the “data fields” (i.e., *operand constructs*) of their documents. Not all of the constructs are easy to build. Indeed, as our subsequent studies show, while datoms can be created with a relatively simple editor (Cabitza et al. 2011b), which we realized to allow users to both create data fields and their templates, operations clearly need to be associated with specific behaviors exposed by the platform or the infrastructure (like printing or sending as a message).

The grammar allows more complex *operator constructs* to be recursively defined by composing more elemental ones by means of suitable *functional operator constructs* and their corresponding *primitives* (see below).

Structures. These are what any bricoleur creates by composing and arranging *constructing constructs* together. We distinguish between *layout structures*[32] and *control structures*. The former ones are sort of material (yet non-necessarily tangible) and symbolic work spaces that are recognized by members of a community of practitioners as the physically inscribed technological artifacts (Orlikowski 2000) with which to carry out their work. In document-based information systems, *layout structures* are the document templates of forms and charts that are to be used to both accumulate data and coordinate activities (Berg 1999), endowed with both physical properties (i.e., the topological arrangement of the constructs mentioned above, i.e., data fields and sections) and symbolic properties (the boilerplate texts, any iconic element and visual affordances conveyed through the graphical interface). For example, in WOAD, a *web structure* is a graph of hyperlinked templates (i.e., *layout structures*); these latter are a set of *didgets*: a didget is a *topological object*, i.e., a datom (see above) that is put in some place, i.e., is coupled to a set of *coordinates* (that in WOAD are represented as Cartesian pairs with respect to the

[32]We prefer the expression “layout structure” instead of “information structure” (or “data structure”), which would perhaps be the traditional mode to indicate those structures, as the latter term would have given the nod to the high-level, conceptual element those structures could be referred to by a human user. Conversely, we mean to hint at the material, spatial arrangement of meaningful signs that “act at the surface” in promoting cognitive processes of sense making and interpretation.

Table 11.2 Operator constructs identified in Cabitza (2011)

Document-based operations	
create	This operation is akin to picking a new empty sheet of a specific template to insert into the folder
retrieve	This operation is akin to picking a sheet from an archive and make it available for other operations
open/read	This operation is akin to getting explicit access to the content of a sheet or instance of artifact
write	This operation is akin to adding some new content to the artifact and accumulating new inscriptions on it
select	This operation is akin to pointing either an artifact (among others) or a specific portion of its content
copy	This operation is akin to putting some content into a buffer memory, like a little pocket sheet
correct	This operation is to be considered different from regular writing but rather similar to striking through some
transmit	This operation is akin to sending either the physical artifact or (part of) its content to an external party
print	This operation regards the physical printing, or copy, of part/whole content of an artifact
officialize	This operation regards the formalization/certification of part/whole content of an artifact
annotate	This operation differs from write in that it is aimed at adding informal or side content: it stands to writing as metadata stands to data
attach	This operation can encompass affixing an external resource to the artifact
cache	This operation regards the saving of part/whole content of an artifact for future use (modifications are still possible)
store	This operation regards the storing of the artifact in some repository, where only an operation of retrieve can take it from
protect	This operation regards the preservation of part/whole content of an artifact from further operations
delete	This operation regards the partial/complete elimination of either an artifact or parts of its content

origin of the template). In the domains of computer-aided design and collaborative drawing/editing, a layout structure can be considered the working space where users arrange command docking bars, symbol stencils, and predefined configurations of elements that must be set up before working on them.

On the other hand, *control structures* specify how the computational engine of the underlying layers of the architecture (see Fig. 11.1) reacts in response to events generated at artifact (interface) level, how this latter acts on the content inscribed therein, and how it interacts with the users during their use of the tool. On a formal level, *control structures* are compositions of *rewriting rules* (see the keyword connector in Table 11.1) that express rewriting systems, a general formalism that can be instantiated, e.g., as rule-based control systems, Petri nets, business process modeling language, or any sort of declarative control construct. In

WOAD, the control structures are called *mechanisms*, i.e., if-then rules whose if-part is a Boolean expression that is recursively defined using the predefined datom names as variables and the operators identified above, all together with the (obvious) constants of the basic types. The then part is a sequence of *operator constructs* that has to be executed on the template or on its inner components. In particular, these operators can change the affordances of an instantiated template to convey information to promote collaboration awareness according to its contents (Cabitza and Simone 2012a). Moreover, mechanisms are composed by the (implicit) OR *functional operator construct*.

More complex *control structures* can be obtained by composing more elemental ones by suitable *functional operator constructs* and their corresponding *primitives* (see below).

Annotations. We consider *annotations* as part of the first-class concepts of a logic of bricolage for their central role in work articulation, knowledge sharing, and mutual understanding (Luff et al. 1992; Cadiz et al. 2000; Bringay et al. 2006; Cabitza et al. 2013) yet at a more informal level with respect to institutionalized (lay-out) structures and to the official content that is accumulated therein during situated practice. To this respect, any form of annotation carried out by practitioners over and upon structures and their content can be seen as a more ephemeral, informal, and more user-driven piece of bricolage, which acts at a different layer with respect to primitives, structures, and content (see dynamics and specificity in Fig. 11.1) but that nevertheless plays an equally important role in making the artifacts in use flexible enough to support invisible work (Star and Strauss 1999). Annotations are then either stigmergic signs (Christensen 2014) and marks attached to the borders of documents, extempore comments, and semantic tags from either domain-specific taxonomies or setting-specific folksonomies or nested threads of both, as we described in Cabitza et al. (2012b): all pieces of a bricolage that hosts informal communication and handover between practitioners, their silent and ungoverned work of meaning reconciliation, and the sedimentations of habits and customs in effective (yet still unsupported computationally) conventions of cooperative work. For these reasons, we believe that any working environment aimed at enabling users to preserve (or even augment) their record-keeping conventions in the digitization of their traditional artifacts should support annotation as a first-class class activity of workers in their natural “ecosystem.” In particular, then, also annotations should be referrable in control structures as we described in Cabitza and Simone (2012a, p. 232) and in Cabitza et al. (2012b).

Primitives. Primitives are basic operations that the platform makes available through the editing environment where the bricoleurs can create both their constructs and their structures. To adopt a pseudo-formal analogy, if constructs are the elements of an alphabet, the primitives can be seen as the composition rules of a grammar by which end users can generate meaningful sentences (i.e., structures). Specific primitives allow users to populate these structures with both content and meta-content, that is, any collaborative annotation.

The running environment executes *operator constructs* (both *functional* and *actional* ones) by interpreting them as more or less complex articulation of *primitives*; these latter, in their turn, are domain-independent functionalities exposed by the platform that have been expressed in terms of lower-level application programming interfaces by IT professionals. In particular, the platform conceives the following primitives (besides the usual arithmetic and logical operations like +, −, AND, OR, etc.): *read* and *write* that represent the conceptual operations at the basis of any computation; *bind*, which assigns constants to variables; *aggregate*, by which to build complex operands from simpler ones; *compose*, to build complex operators in terms of functional composition; and *place*, to associate an operand with some coordinates to create a topological object. Moreover, the *annotate* primitive allows us to associate a *domain value*, a *multimedia text*, or a *conventional symbol* (e.g., any character, word or iconic mark) to any construct or to an existing annotation. It is noteworthy that operator constructs can be just domain-specific specializations of *primitives*, like a user-defined procedure *sum()* that can specialize the + primitive of a programming language.

In WOAD, the basic operations conceived for the definition of both templates and mechanisms, i.e., *layout* and *control structures*, respectively, can be related to the *primitives* mentioned above especially *aggregate*, by which more complex datoms (as operand constructs) can be built from simpler ones, and *place*, by which users can associate a *didget* with a Cartesian coordinate with respect to the origin of a given template. A *mechanism* is a simple *control structure* made of a single rewriting rule while the *compose* primitive is a simple OR *functional operator construct*.

As mentioned in the previous section, the *primitives* are offered through an editing environment where *constructs* and *structures* can be defined. For example, in WOAD, this environment is constituted by two visual editors: one for the construction of mechanisms (Cabitza et al. 2012a) and one for the construction of datoms and, by arranging these latter topologically in terms of didgets, templates. The editing environment is associated with an execution environment that has been tested on realistic examples taken from the healthcare domain (Cabitza and Gesso 2012).

The conceptual architecture that is depicted in Fig. 11.1 incorporates that in complex socio-technical systems, change must be expected: indeed, it encompasses different layers that account for both different scopes (see Specificity), concerns (see Task), involved roles, and dynamics. We borrow again some terms from Lanzara's logic of bricolage to qualify the changing rate of the layers constituting the conceptual architecture. Any form of annotation carried out by practitioners over and upon structures and their content can be seen as a more transient, informal, and more user-driven piece of bricolage, which acts as a sort of different layer with respect to content, structures, constructs, and obviously the platform's primitives (see Fig. 11.1); nevertheless (or right in virtue of this complementarity), annotations play an equally important role in making the artifacts-in-use flexible enough to support also invisible work (Star and Strauss 1999) and hence fully appropriated by their users.

Layout structures are the immediately less transient components as they correspond to working spaces that users flexibly accommodate to their changing needs (Harris and Henderson 1999). Decoupling layouts, i.e., the data structures, from the logic acting upon them, i.e., the control structures, is a well-known engineering principle that the framework recognizes. However, although formally decoupled, control structures in practice follow and support the modifications of the objects/artifacts to which they are applied: therefore, they are at the same level of change rate.

On the other hand, constructs, especially the more basic and atomic ones, can be considered as changing at a slower pace than the structures they are part of, while the more complex ones can be revised (modified, deleted) more often but probably less frequently than the above mentioned structures. Primitives, conversely, are almost persistent: informally speaking, the composition rules must be more stable than the objects they apply to. Finally, the layers of the underlying technological infrastructure can be considered as almost permanent, in that changes can be planned, postponed, and made incrementally depending on the triggering technological evolution or organizational strategy. The emphasis on this aspect is motivated by the fact that IT professionals have to build the infrastructure and its relationships with constructs and primitives plastic enough to avoid any friction between the different layers that "drift" at different speed according to regular technological evolution and the users' needs.

11.5.2 Some First Implications for Research

The three-layered architecture described above is aimed at addressing the user-centered requirement to provide shop-floor practitioners involved in a digitization program with (at least) the same space of possibility they have when they work with non-digitized artifacts. In so doing, end users would get the opportunity to transition from their paper-based artifacts to computationally augmented ones by means of the editing and working environments, so that the layout structures that scaffold their activities (Orlikowski 2006) would change with the necessary gradualness (or do not differ at all) at least in theory.

To this aim, the grammar is left purposely flat, general, and simple: we do not want to introduce surreptitious entities, like the concept of artifact, activity, task, role, and the like, which traditional methods of software production may already employ as primitive elements for the phase of design. We have already recalled how any design of IT technologies either produces or adopts a model, sooner or later. These models, irrespective of the layers at which they manifest or are adopted, will necessarily end up by conflicting with work practices (for a recent account on this phenomenon see, e.g., Morrison et al. 2011). This is because these practices "by definition" change over time and make sense only in their doing (see Sect. 11.2), while models equally "by definition" introduce the level of representational rigidity that is necessary for their role in requirement elicitation and formal specifications (e.g., Bowers 1991; Robinson and Bannon 1991; Bannon 1994).

However, the architecture depicted in Fig. 11.1 requires a radical change of perspective for all the stakeholders involved in technology conception and construction. In particular, this proposal requires us to focus on the idiosyncratic and fine grained ways in which users cope with unexpected change in their work environment (Bannon 1992). For IT professionals, this means focusing on how constraints have to be dynamically expressed to support the definition of the appropriate ordering of action and interaction in cooperative work in any circumstance (e.g., Pesic et al. 2007; van der Aalst et al. 2009), which pieces of information are used to support articulation and cooperative work, how these are arranged in suitable artifacts (e.g., Nemeth 2003; Cabitza 2011), and what habits, customs, and conventions are at stake and silently inform the exchange of information and the sense making occurring within and across communities of cooperating actors (Mark 2002; Cabitza et al. 2009). In sum, conceiving artifacts (and entangled tasks) as more or less transient "entities" emerging from the composition of constructs requires to understand what elementary bricks users *already* have on hand to flexibly compose their artifacts[33] and to conceive of ways to make those bricks computational, that is, associated with specific system behaviors (or functionalities), so that the performative and entangled nature of tasks and artifacts can be preserved and supported.

With respect to a research-oriented agenda, this requires further studies and meta-studies in the same vein of those by Martin and Sommerville (2004) and Cabitza (2011), which aim to identify recurring and "universal" elemental operations/behaviors. Their identification would facilitate the reuse of ways to map either domain- or setting-specific (operator) constructs with the APIs that the common platform has to expose to make the execution of *operator constructs* possible. These studies would share the assumption that leveraging general *operator constructs* will not impose users any specific practice or way to treat information (which is mainly represented in terms of *operand* constructs). This assumption seems reasonable first because the identified operations would be intended to be as "atomic" and therefore as elementary as possible and second because what could change in any specific setting would be the practices users are familiar with and hence the way users would make sense of and articulate together those basic elements within their work.

11.5.3 For Whom Tolls the Bell?

In the previous section, we hinted at the fact that an architecture that enables bricolage requires all the stakeholders to reconfigure their traditional roles to make the best use of it within the win-win game that motivates such reconfiguration. But who are the stakeholders that are involved on a practical level? In this section, we will just limit ourselves to the ones that are closer to the task-artifact entanglement[34]:

[33]We recall here the requirement that bricoleurs already *know* the available pieces (see Section 3).

[34]We are aware that buyers, top management executives, middle management officers, more or less official and institutionalized representatives of business units, and their employees have always been part and parcel of the development process of a corporate information system. However,

traditional frameworks usually denote these as end users, key users, and actors from one side of the divide and designers, analysts, programmers, and developers from the other.[35]

The former actors are those that are supposed to invest an important effort in bricolaging with the system, in the hope that this could pay off in terms of a better fit between the resulting system and their needs and of a smaller impact on their traditional coordinative practices and accepted power relationships. In this regard, it is often argued that one should distinguish at least between the regular end user and the so-called power user. This distinction, which can be of some value for purely analytical purposes, should yet be treated with caution if it is to drive the decision of "what to offer/allow to whom." The conventional label of power user, far from being used – as often is – to indicate a role having special rights in modifying the technology (like a sort of administrator, who is distinct from regular users for her "powers"), should be rather interpreted as originally intended by Bandini and Simone (2006), i.e., as an organizational or even more informal category that allows us to distinguish end users on the basis of their motivations in improving the artifacts *they also use* and for the competences they have acquired in understanding how things could be changed and why. In this light, power users are not the "chosen ones" that receive the right to modify the application from above but rather who, in virtue of their motivations and competencies, are either formally or informally delegated by their colleagues with the aim of taking personal care that the tool continuously evolves and fits the current needs of the community where it is put to work.

Therefore, within our perspective, the difference between power and regular user fades away in the notion of bricoleur: someone that can be factually involved in constructing and developing the bricolage or that anyway uses it and hence contributes in building and consolidating related habits and conventions of usage and interpretation. For this reason, access to the editing environment should be purposely left to be regulated according to social control and local and socially relevant conventions and initiatives that are just outside the scope of the technology itself.

In regard to the IT practitioners, obviously abandoning the traditional view of rational design does not entail getting rid of designers at all; besides being socially impossible, this sounds also undesirable. Rather, it requires designers, business analysts, and IT analysts to focus on different initial purposes and services to supply, as we hinted at in Sect. 11.5.2. In Sect. 11.3, we recalled the work by Hartswood et al. (2000) and hinted at the role of designers in terms of *facilitators* of the process of co-construction of both tasks and artifacts. A similar role has been identified by the approach that proposes participatory evolutionary design as a virtuous integration

---

articulating the reconfiguration of the larger actor network that encompasses all these levels of involvement and accountability would be out of the chapter's scope.

[35]Cabitza and Simone (2012c) show that this divide has historical roots, and hence it is contingent. In particular, the "divide" took place approximately in the second half of the 1950s when the computer, which had been thus far intended only as a mathematical instrument for which each of its users had to write his/her own code to be executed when it was his/her turn, became a full-fledged time-sharing equipment and established itself as a business machine or better yet an electronic data-processing machine (O'Neill 1992; Campbell-Kelly and Aspray 2004).

of EUD and participatory design (Sumner and Stolze 1997). The term “facilitator” yet must be taken more in the connotation first discussed by Hirschheim and Klein (1989), that is, more as that of a *catalyst* within a chemical reaction that really follows unpredictable and, above all, uncontrollable dynamics. Otherwise, the risk is to conceive them as professionals supposed to facilitate the process in which computer-based systems are finally accepted as “perfect bureaucratic tools” (Harris and Henderson 1999) and adopted within a community of practice or organizational setting.

In this view, we can detect two main roles involved in IT development from the IT perspective, which are characterized by specific and complementary competences. One, who acts as the catalyst/facilitator mentioned above, could be referred to as a *maieuta*-designer. Although such a word would be pronounced quite similarly to that of meta-designer,[36] its meaning would refer to a quite different thing. Maieuta is one who performs the art of maieutics, the Socratic approach where someone helps *bring out* implicit notions in the interlocutors’ beliefs, mainly through a dialogic and narrative way encompassing a series of open questions that do not necessarily require an answer,[37] or just help them further refine their understanding and become more autonomous in their expression. Such a designer is primarily concerned with the front end of the enabling technology, i.e., with the graphical and semiotic aspects of the artifacts “to be built and to be used” through it. Also, the maieuta-designer is supposed to “close” the merely technological part of the design process and to *pass the baton* on the end users, i.e., (the bricoleurs), by helping them to find the means and motivations for the “in vivo” development of their artifacts on their own. As such, the maieuta-designer does not have to possess strong programming or architectural skills: rather he/she has to be a domain expert and a connoisseur of how the users of a particular domain (if not particular setting) are used to conceiving their tools and tinkering them over time (to what ends, on the basis of what political and cultural drives and constraints, and the like) in order to help users in exploiting the available editing environment in such a way that their bricolage does not become an erratic process but rather is *sustainable* over time.

Typical technology-oriented competencies must be conversely mastered by the IT professionals, or back-end designers, working on and developing the platform itself: these are called to the role of guaranteeing that the artifacts built on top of the platform can evolve over time, that is, that the enabling environments are easy to use for as many actual users as possible. This means guaranteeing that the best software engineering techniques (e.g., modularity, integration of data and routines) are employed to make the platform and exposed environments powerful and flexible enough to allow for the bricolage activities at the higher-level layers of the overall architecture, besides guaranteeing also that the platform is modular and robust enough to cope (and align) with (low rate) changes in the underlying infrastructure. We could say that “designing for unanticipated construction” could flank (or perhaps substitute) the old claim for “designing for unanticipated use” by Robinson (1993).

[36]And this is not completely by chance: mee’yootah vs. ’mee-tah.

[37]A list of this kind of questions can be found in Cabitza et al. (2014b).

11.5.4 What Is Outside Like This?

Our proposal tries to reinvigorate a discourse among the current mythologies of system design that conceives the progressive delegation of power and control to end users as a feasible way to cope with increasingly complex socio-technical systems, supported by increasing complicated IT systems (Latour 1996).

This goal can be achieved by supporting two dimensions of EUD: the collaboration among end users in the construction and tailoring of their applications and the technical aspects of the construction itself. The former dimension is captured by the notion of "use discourse" discussed in (Pipek and Wulf 2009): this concept has shaped the construction of environments that support users in negotiating the configuration of the software infrastructures and that are characterized by typical collaboration tools such as discussion forums, representation sharing (among which process diagrams) and annotation, and voting systems (Wulf et al. 2008; Stevens 2010).

As an alternative approach, in recent years, a small number of frameworks have been developed to support the end users as bricoleurs in the sense discussed in the previous sections. Among these, we obviously include WOAD, the document-based information system platform that we have introduced in Sect. 11.5.1. Although WOAD has been natively conceived to allow for the degree of flexibility and user autonomy, the specialist literature reports also other platforms and frameworks that exhibit similar features. For instance, placeless documents (Dourish et al. 2000) introduced the idea of document properties that are attached by single end users and, above all, properties that represent active ways to operate with documents (called active properties): users can add these properties to documents to make them carry executable code that can be invoked to control or augment their functionalities. This work, to our knowledge, was among the first to carry into the HCI arena notions from prototype-based object-oriented programming and operating system programming, like attaching code to documents as a means to control their behavior and the idea of letting users develop some bunches of runnable code to extend the system functionalities. WOAD decouples layout structures from control structures, although these can be related to each other by means of if-then mechanisms defined over the document's content.

Enabling end users to build their own documents is a common trait among recent initiatives on visual data-driven form generation; these projects are usually aimed at allowing users to generate even complex forms, intended as data-entry points to an underlying flat data structure, without particular programming skills, e.g., by means of a visual editor like Microsoft® InfoPath® as described in Mamlin et al. (2006) or by means of the layout mode in FileMaker Pro® as used in Chen and Akay (2011). This allows us to take "form design out of the programmers' hands and put it into the realm of content management, much as form-generation tools (like Ruby on Rails or Plone's Archetypes) aid the developer in rapidly generating forms" (Mamlin et al. 2006), and hence to address a specific need that so far has been raised by practitioners in the healthcare domain (e.g., Mamlin et al. 2006;

Morrison and Blackwell 2009; Chen and Akay 2011; Cabitza et al. 2011a). The system described by Mamlin et al (2006) presents also the feature to associate form elements with specific rules (expressed in the Arden syntax), making this similar to WOAD, although the editor defined in this latter framework allows for the reuse of form components (i.e., the datoms) and for the abovementioned decoupling between layout and the logic-based control flow of execution (datoms vs. mechanisms).

Finally, some effort has been paid by researchers to address the requirement of making the end user really autonomous in creating their document-specific rules, and this is usually enabled by means of visual and user-friendly tools (e.g., Cabitza and Gesso 2012; Krebs et al. 2012). An even more comprehensive approach to this general aim was proposed by Harel and Marelly (2003) in the Play framework: this latter allows users to build reactive systems by playing, so to speak, their specifications in a performative way, that is, through scenarios that are subsequently implemented by means of a Play-Engine that "plays out" the corresponding models of interaction; these are explicitly represented in terms of multistep control structures, what the authors call "live sequence charts." These are hence more complex and articulated interaction structures than simple rules are. Whether they cope well with unknown emergent behaviors is less certain. However, in Play, users can specify these models of human-computer interaction in a way that is innovative and peculiarly aligned with some of the tenets we discussed above: end users can interact with prototype user interfaces and have the system build the corresponding structures or write the intended behavior and its main exceptions handling procedures in brief sentences expressed in natural (yet structured) language or even tell it directly to the system, in a sort of versatile multimode way to teach the system what to do if some events occur (typically at interface level).

The Play framework has been specifically proposed as a concrete first step toward what Harel (2008) suggestively calls the *liberation of programming from its three straightjackets*: these are the "1) need to write down a program as a symbolic, textual, or graphical artifact; 2) the need to specify requirements (the what) separately from the program (the how) and to pit one against the other; 3) the need to structure behavior according to the system's structure, providing each piece or object with its full behavior" (p. 29). The aim of liberating programming from these representational straightjackets resonates in very close affinity with the tenets of an EUD approach and seems to go in the direction of drawing a common agenda where practitioners from different disciplines can perhaps meet together and inform their own research and development initiatives in a positive manner.

11.6 Some Final Remarks for Future Discussion

In this section, we will just outline two important aspects that should be the object of future research (or discussion) on the concrete applicability of the *laissez-faire* method and of the *logic of bricolage* in the development and evolution of socially embedded systems. These two topics relate to important strands of research that are receiving more interest from diverse communities of researchers involved in studying the impact of IT in social settings in the recent years. We broadly denote these two topics as "concerns about risk" and "concerns about interoperability" (and hence standardization).

11.6.1 How Risky Is a Different Development Strategy?

One could detect an irony in our advocacy of a *laissez-faire* approach for the construction of an efficient, effective, and safe technology, whereas it has been the need to guarantee such qualities that motivated the consolidation of engineering methods and methodology in IT system construction (Haigh 2010; Cabitza and Simone 2012c). This feeling could be reinforced by our explicit confidence that an environment enabling and supporting bricolage by end users could be a feasible alternative to any *-design of those systems. Indeed, in the specialist literature, there is a tendency to consider bricolage as akin to improvisation (Weick 1993; Lanzara 1999) and the bricoleur as someone, at best, who draws on the materials at hand to create a response to a task on the spot (Levi-Strauss 1966); in Lanzara's words: "in a broadly diffused engineering ideology, bricolage is usually associated with second- best solutions, maladaptation, imperfection, inefficiency, incompleteness, slowness." This prejudice is repeated several times within the system development discourse.

One reason for that is to be found in the misunderstanding coming from understating the radical novelty that the myth of the bricoleur carries. In fact, as long as bricolage is evoked in discourses that still refer to a traditional way to build computational artifacts or only moderate it (e.g., design with users, meta-design), that is, as long as bricolage is ingrained in a traditional way to think of IT design (even in the light of contributions coming from participatory design, action research, and ethnography), we will keep undermining the original sense of this concept in some (important) way and, worse yet for our aims, weakening its potential to move into a new and more useful mythology.

From what we have argued at length, it should be clear that we stress the ability of the bricoleur to "work and play with the stock [with] parts that are not standardized or invented, [but rather] appropriated for new uses" (Weinstein and Weinstein 1991, pp. 161–162) and that are taken from "an inventory of semi-defined elements [that] are at the same time abstract and concrete [and that] carry a meaning, given to them by their past uses and the bricoleur's experience, knowledge and skill, a meaning which can be modified, up to a point, by the requirements of the project and the bricoleur's intentions" (Louridas 1999): what we above called constructs. In this compositional mythology then, the concept of *bricolant* bricoleur should not be seen any longer as an "improviser," a tinker, hobbyist, or hacker in the negative connotation of these terms (e.g., Ciborra 2002, p. 47–48) but rather as a creative actor who is enabled by the sponsors of the digitization initiative to reach their purposes in the awareness that only end users are competent *enough* to reach a sustainable balance between effectiveness and efficiency in cooperative ambits and, above all, to meaningfully improvise in the face of the unexpected, in virtue of the fact that users, communities, and organizations *already* exist (i.e., they preexist digitization and informating initiatives) and *already* possess "a disposition towards their environment [. . . ,] already [are] committed in a self-meaningful manner towards [their] own survival and prosperity" (Angell and Ilharco 2009).

Supporting bricolage then is not the slothful retreat of the blasé researcher that releases responsibility by advocating a more empowered and active role of end users in virtue of her democratic feelings. Rather it is the ultimate strategy to make the complex socio-technical systems that our IT solutions contribute more *resilient* in face of the *normality of accidents* (Perrow 1999). More than this, we submit that an architecture that adopts a *laissez-faire* method and the *logic of bricolage* described above could be said to be both intrinsically resilient and *evolutive*, two terms that are attracting more and more interest by researchers involved in the safety of IT systems (Hanseth and Ciborra 2007) especially in critical or delicate settings, like healthcare (Hollnagel et al. 2008). These two concepts engage the capability of a system to react or change to unexpected events, changes, or conditions at different time scales, respectively, short and long with respect to the occurrence of unexpected event. Being both resilient, that is, able to reach a stable and safe working state after that some unexpected event has occurred, and evolutive, that is, able to grow and increase one's own fit with respect to the surrounding environment, is a fundamental characteristic of socio-technical systems to properly face the increasing odds of failure of some of their multiple components that can result from the increasing complexity of their interrelated parts and corresponding links. This characteristic also constitutes progress with respect to the concept of robustness which refers to the capability of a system to resist and withstand adverse events and change, also in virtue of design and analysis phases usually aimed at identifying, prioritizing, and handling specific exceptions or at reducing the opportunity for their occurrence.

The architecture we envision above is intrinsically resilient as it, paradoxical as it may seem, delegates to end users the burden and responsibility of reacting to unexpected events by leveraging their innate creativity and the invaluable, and often irremediably tacit, knowledge of the overall system dynamics, sometimes much more based on intuition than on rationality (Mark and Semaan 2008) and because such an architecture purposely avoids providing users with information and execution structures that are constrained and articulated on the basis of strong assumptions on how the system will behave under certain conditions: a bricolage-based information system is just an environment for both the human and automated manipulation and processing of signs.[38] Moreover, interactive systems that are built on top of such an architecture are naturally and concretely open to evolution as they, differently from those systems that are built by someone else than their actual

[38]If the worst occurs, e.g., if power goes down, the overall socio-technical system is made more resilient simply by printing out some layout structures on paper and having the users work as usual, just without the computational augmentation of those structures.

users, are changed opportunistically by end users on their own when (or in a short time after that) they feel this necessary to accommodate their artifacts and tools to emerging conditions and newly recurring situations. Again in Lanzara's words:

> [...], systems assembled by bricolage have an evolutionary advantage: being loosely connected and incoherent assemblies of mixed components, they can be partially reworked without much investment effort. Bricolage is a design strategy that makes sunk costs recoverable. In case of system's depletion, obsolescence or low performance, regeneration can be done without having to throw away the whole structure. [...] As a consequence, systems are persistent and robust because cannot be changed or moved easily, but at the same time keep structural plasticity and exhibit some self-correcting properties. Innovation can be accommodated locally. [...] As [bricolage] exploits the properties of existing structures for interactive and generative purposes, it successfully mediates the dilemmas of change and stability, innovation and conservation. On the one hand, by experimenting with transient constructs it allows for some variability and improvisation without incurring in the possible disruptions caused by excessive instability and radical change; on the other hand, by assembling robust but furtherly manipulable structures, it allows for some order and reliability without curbing the chances for system improvement and innovation. In short, it makes both radical innovation and complete unraveling unlikely. (p. 347)

We subscribe to this understanding of the role that an active, conscious, and responsible *bricoleur* can play in system development but also associate this with the awareness that new platforms must be built supporting this role and new professionals must be educated to mediate between the possibly conflicting stances of such an empowered actor and other roles that are in a hierarchical relationship with it.

11.6.2 Interoperability Concerns

The second issue we propose for future research and discussion regards the tension between the global dimension and the local one in the construction of a technology aimed at supporting whole organizations and/or networks. This is therefore the extension of what we have discussed so far in terms of collaboration within groups of limited size, what we call the local dimension, in the new terms of interoperability "in the large," i.e., across multiple settings and organizations. The separation between global and local has been already shown as illusory by who have recently proposed the concept of information infrastructure (e.g., Fitzpatrick and Ellingsen 2012), as it is recognized that any local event or practice can have the potential to affect the overall system, sooner or later.

In Sect. 11.3, we very briefly hinted at how this separation can be just seen as one way by which the interests of someone (typically institutional authorities of control and high-level regulatory bodies at regional or national level) are imposed on the practices of others (typically the producers of inscribed data and their first consumers for articulative reasons (Berg 1999; Winthereik and Vikkelso 2005)). Consequently, addressing the tension between local vs. global requirements means to also address one of the main root causes behind the pervasive (and still relatively neglected) phenomenon usually denoted as "work-around."[39] Such a circumvention of the system and its intended (and designed) uses that users perform to overcome the rigidity that the global view of IS imposes on their local practices can indeed undermine any serious attempt to engineer and deliver safe and robust technologies in socio-technical systems (Niazkhani et al. 2011; Handel and Poltrock 2011). In this section, we will outline how a laissez-faire method can be compatible with the increasing need for global interoperability between local systems and, in its little own, can contribute in going beyond the abovementioned tension by interpreting such seeming opposition in a more dialectical way toward the development of information systems that can be flexible enough to meet the local needs of data producers', i.e., the primary users of any information (Cabitza and Simone 2012c), as well as the needs/expectations of some relevant data consumers (i.e., secondary users) at the same time.

The relationship between the applications or functionalities that are developed by end users to respond to their local needs and the information systems that are conceived at the global level of the organizations has been investigated by Doerner et al. (2009) by leveraging the component-based approach (specifically an evolution of the service-oriented approach) and prefiguring an evolution of ERP systems that increases the role of end users as "prosumers" thanks to the transformation of the "cathedral" (i.e., ERP system) into a "bazaar" where end users can exploit the flexibility of SOA architectures and cloud computing. With a similar perspective, the contribution by Beringer and Latzina in this book illustrates the concept of transformative user experience (TUX) "as a design approach in order to dissolve the boundaries between applications and, to transform aspects of appropriation [. . .] into an intrinsic capability of the infrastructure rather than a post-design phenomenon inferred from observing how users interacted with the system. [. . .] One way to achieve this flexibility is to underspecify the user interface of an application to allow users to impose different meanings and usages on an IT-artifact at use time. [. . .] With a transformative user experience approach, we can overcome [the] concept of (unexpected) adaptation, and foresee ostensive usages by enabling transitions from one meaning to another in the system environment." With this aim, they introduce the notion of "elasticity" as a feature of both "containers" (that identify dynamic working spaces) and of the objects they contain: elasticity makes them able to take different meanings according to their dynamic contexts. On the one hand, this proposal aims to overcome the rigidity of the component-based approaches to EUD; however, on the other hand, it claims that "In a corporate work environment, those transitions happen most typically between process-oriented backends and task-oriented front ends," thus linking elasticity to a predefined set of contexts and therefore limiting the degree of tailorability of the resulting environment.

As aptly recalled by Lanzara (1999), "anthropologist Clifford Geertz has pointed out that the more we try to make the world 'global,' the more the world responds with the emergence of multiple 'local worlds' and identities that seem to be

[39] A short literature review of this concept can be found in (Cabitza and Simone 2013b).

irreducible to one another." In the same line but with a more technological perspective, Ciborra (1992) suggests that "Top management needs to appreciate local fluctuations in system practices as a repository of unique innovations and commit adequate resources to their development, even if the systems go against traditional approaches. Rather than looking for standard models in the business strategy literature, [strategic information systems] should be sought in the theory and practice of organizational learning and innovation, both incremental and radical." We believe that this passage offers an interesting stimulus to go beyond the way in which firms conceive their (strategic) information systems. Actually, letting information systems "[. . .] emerge from the grass roots of the organization, out of end user hacking, computing, and tinkering" asks for a significant change of perspective in IT design that is closely related to the performative and bricolage-oriented stance we advocated in this chapter. Since "organizational learning and innovation" occur where practices are and action is (Dourish 2001), we subscribe the suggestion by Ciborra (1992) and Lanzara (1999) to start from this "local dimension" to build a technological support that promotes firms' vitality not only for the sake of innovation but also for an effective every day performance. Moreover, since "learning and innovation" are "both incremental and radical," the requirement of having different layer dynamics (or change rate) that was discussed in Sect. 11.5 can be put in relation with the need to cope with incremental and radical changes in the organization itself and in the coevolving technology.

Our suggestion is to regard the "whole" not as a centralized and monolithic entity (to some extent) but rather as the composition of "small" entities, highly specialized to the local needs and highly connected in a web of loose connections linking "local spaces," i.e., peer nodes that are each characterized by local structure and semantics (Bandini et al. 2007) and that are easily adaptable to unexpected contingencies since they are concerned with (and hence control) a limited and well-known "piece" of the world. Obviously, a need to make this new kind of "whole" coherent and consistent with the overall good performance of the network would arise, which is the main concern of any higher-level management unit. However, instead of having the management promote (and enforce) coherence in terms of "obtrusive" control of local behaviors and top-down ontologies to order resources, the management unit itself might behave as a "local entity" with specific goals and expectations on data produced by other units, expressed in terms of its own local quality level constraints. These latter can be exposed as public expectations that other units can (or have to) comply with to interoperate, once either the producer or the consumer have selected what data structures are involved by these constraints and how these data have to be arranged and aggregated to respect the consumer's needs.

In this scenario, interoperability is achieved not by semantically enriching data on the producer's side, i.e., where those data are not natively attached with that semantics, but rather by having consumers pragmatically express what data they need, subscribe to a set of data that are exposed by producers, and, possibly, express also how they need those data be presented (i.e., reported, typically in aggregated form). In this view, the definition of "standard structures" (which play the role of boundary objects (Star and Bowker 1999)) end up by regarding much simpler pieces of information, i.e., a subset of the operand constructs of one unit that this latter exposes to the outside world (or to specific correspondent units, typically of higher level in a social hierarchy). As said above, these latter items can be considered as changing with a low speed rate, since they play a sort of minimum data set that characterizes the unit at hand, but they are in any case easily modifiable since they are not universally (and hence not rigidly) incorporated in more complex structures. In this view, consistency has to be obtained by an effective monitoring of the received structures, rather than by a prescriptive way on how to achieve interoperability about them.

11.7 Conclusions

Let every careful man be very far from writing about things truly worthy of care. (Plato, 352 BC, 7th letter)

The main claim of this chapter is that, in order to bridge the gap between what users need and what is given to them as solutions to those needs, the concept of design has to be substantially challenged and its role in IT development reformulated: in other words, that the IT systems should be at least partially *de-designed* (Cabitza 2014). To this aim, we submit that an old mythology of design, which is based on the separation between conceptual design and situated use and consequently on the modeling activity that entails this separation, should be abandoned in favor of a new mythology. We advocate this new mythology be grounded on both the notion of performativity, from the conceptual perspective, and on the notion of (bricolant/actant) bricoleur from the more practical perspective.

Reviewing the main tenets of this mythology has brought us to introducing a lean method for the development of socially embedded technologies, epitomized by the motto "*laissez faire les bricoleurs*" and the preliminary proposal of a "*logic of bricolage*" that specific environments should enact to empower end users in the process of development of their tools.

Quite distinct from those who welcome the increasing blurring of the roles of designers and users (e.g., Fischer et al. 2004; Johannessen et al. 2012), we do not advocate that users should increasingly "act as designers" (and researchers work to that aim). Rather, we rather believe that such an idea would foster an approach by which users adopt a spurious attitude. There is a danger that users become people who think to "design her own practices" as well (as claimed recently by Johannessen et al. 2012). Conversely, the role of IT professionals and of end users has to be characterized by a clear separation of concerns in the development of computer-based supports of cooperative, organizational work: hard engineering-based *design of meta-systems* for the former ones and *bricolaging* for the latter ones. Indeed, we submit that this separation is at least conceptually (if not also pragmatically) opposite to proposals that advocate a tight integration between, if not a unification of, conceptual design and end-user practices (e.g., the participatory design and the meta-design frameworks). This separation can have the advantage of making the relationships between these two roles not only less harmfully ambiguous but also and above all more productive with respect to the timely and effective deployment and maintenance of computational artifacts, since end users would be in full control of this process.

As we have mentioned in Sect. 11.4, there are examples of technologies that can be seen as steps toward the goal of letting users be in true control of the technology they need and wish to use. At a cursory glance, these technologies implement different kinds of platforms that enable users to perform bricolage-like activities in which the pieces that these platforms make available are composed and arranged in meaningful ways. However, there is still a long way to go in order to collect findings that would confirm that there is an alternative way to design than the conceptual and representational one that is currently ruling in our development methodologies and professional practices.

Apart from any technical consideration, this new approach would require a substantial change in the way young IT-related students are educated for information system design: this is traditionally based on a plethora of data models, from the business-oriented one (the conceptual model) up to the more machine-related one (the physical model), and on a collection of business process models and notations by which to describe how work is (or should be) carried out on those data. However, from a broader perspective, the main role that we have advocated for the development of collaborative applications and information systems, namely, the role of the facilitating *maieuta*-designer, would instead require an educational agenda that is quite different. This would encompass, for instance, teaching the basics of social informatics (Kling et al. 2005) and semiotics (de Souza and Leitao 2009; Beynon-Davies 2011), some qualitative research methods and techniques aptly adapted to the IT domain (Kling et al. 2005) and insights on current theories on IT impact on socio-technical systems and on both change and risk management (Hanseth and Ciborra 2007), as well as notions of socially informed history of technological evolution (Akera and Aspray 2004) and its interpretation from the humanities (Oudshoorn and Pinch 2005). The point is that all three of the roles we discussed in Sect. 11.5.3, namely, the user (as expert of the setting), the facilitator (as domain expert), and the IT developer (as expert of infrastructural concerns), should receive a newly formulated or seriously revisited educational program so that an effective way to take "human actors" seriously can be promoted again (Bannon 1992).

On the other hand, the layered conceptual architecture that we have illustrated has still to prove its practical value in a reasonable range of application domains (or settings) where legacy systems do exist and cannot be "obliterated"[40]: our personal research experience makes us confident that such an architecture is promising for the case of document-based, knowledge-intensive collaborative (information) systems; although many such systems can be found in the world out there, we are aware that this macroclass of applications simply does not cover all IT-based supports.

[40]To make a very long story short, legacy systems that automated data structures can – and should – be preserved and wrapped as new local nodes of the network described in Section 6.2; but what destiny to give to those legacy systems that once "automated" procedures and workflows . . . ?

In any case, in order to go a step further in this direction, better platforms and environments supporting an effective and reliable EUD approach are needed. We have proposed some basic principle on which these systems could all be based on: decoupled modularity of information and control structures, loose integration of the latter ones in terms of recomposition of elemental common constructs according to local needs (as opposed to the construction of unifying general schemes), full homogeneity across the layers for the construction of aggregated functionalities so that users can access them and operate with them with the same high-level language, and finally, tools supporting the technology development and managing its intrinsic complexity that are based on users' building practices (vs. the introduction of more or less "hard" engineering tools in EUD).

The two brief (and necessarily partial) outlines of the current discourses on performativity and bricolage, as well as our discourse, encompass notions like the *task-artifact entanglement*, the requirement of *universatility* for EUD environments, the concept of the (*bricolant and actant*) *bricoleur*, the foundation of a *logic of bricolage*, the distinction between *maieuta*-designers and traditional designers (also on an educational level), the *laissez-faire method* as a purposeful way to cope with the complexity of any socio-technical system and de-design its "technical" component, and even the scant peek to the *local/global illusion*; these are all notions provided as pieces of a bricolage. Like any bricolage, we do not see a particular truth in any of the pieces we brought together in this chapter; rather we have argued about the potential of the resulting jigsaw puzzle, in our opinion coherently kept together by the mythology of the performative end users, as a whole to come out being useful. Obviously our hope is that the EUSSET forum will host many similar discourses and give them some sort of legitimacy to inform common initiatives of research, education, and IT professional practice in the near future.

References

Akera, A., & Aspray, W. (Eds.). (2004). *Using history to teach computer science and related disciplines*. Washington, DC: Computing Research Association.

Anderson, S., Hardstone, G., Procter, R., & Williams, R. (2008). Down in the (Data)base(ment): Supporting configuration in organizational information systems. In *Resources, co-evolution and artifacts* (pp. 221–253). London: Springer. Retrieved from dx.doi.org/10.1007/978-1-84628-901-9_9

Angell, I., & Ilharco, F. (2009). Dispositioning IT all: A theory for thriving without models. In *Bricolage, care and information Claudio Ciborra's legacy in information systems research* (pp. 401–422). London: Palgrave Macmillan.

Ardito, C., Costabile, M. F., Matera, M., Piccinno, A., Desolda, G., & Picozzi, M. (2012). Composition of situational interactive spaces by end users. In *NordiCHI 2012: Proceedings of the 7th nordic conference on human-computer interaction: Making sense through design, October 14th–17th, Copenhagen, Denmark*. ACM Press.

Ash, J. S., Berg, M., & Coiera, E. (2004). Some unintended consequences of information technology in health care: The nature of patient care information system-related errors. *Journal of the American Medical Informatics Association, 11*(2), 104–112. doi:10.1197/jamia.M1471.

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

Bandini, S., & Simone, C. (2006). EUD as integration of components off-the-shelf: The role of software professionals knowledge artifacts. In *End user development* (Vol. 9, pp. 347–369). Kluwer Academic Publishers.

Bandini, S., Sarini, M., Simone, C., & Vizzari, G. (2007). WWW in the small towards sustainable adaptivity. *World Wide Web, 10*(4), 471–501. doi:10.1007/s11280-007-0024-y.

Bannon, L. (1992). From human factors to human actors: The role of psychology and human-computer interaction studies in system design. In J. Greenbaum & M. Kyng (Eds.), *Design at work* (pp. 25–44). Hillsdale: L. Erlbaum Associates Inc. Retrieved from dl.acm.org/citation.cfm?id = 125470.125458

Bannon, L. (1994). CSCW, challenging perspectives on work and technology. In *Proceedings of the "Information Technology & Organisational Change" Nijenrode Business School, The Netherlands, 28–29 April, 1994*.

Barad, K. (2003). Posthumanist performativity: Toward an understanding of how matter comes to matter. *Signs: Journal of Women in Culture and Society, 28*(3), 801–831. doi:10.1086/345321.

Bath, C. (2009). Searching for methodology: Feminist technology design in computer science. In *Proceedings of the 5th European symposium on gender & ICT digital cultures: Participation - Empowerment - Diversity, March 5–7, 2009, University of Bremen, Bremen*.

Beath, C. M., & Orlikowski, W. J. (1994). The contradictory structure of systems development methodologies: Deconstructing the IS-user relationship in information engineering. *Information Systems Research, 5*(4), 350–377. doi:10.1287/isre.5.4.350.

Berg, M. (1999). Accumulating and coordinating: Occasions for information technologies in medical work. *Computer Supported Cooperative Work (CSCW), 8*(4), 373–401.

Beynon-Davies, P. (2011). *Significance: Exploring the nature of information, systems and technology*. New York: Palgrave Macmillan.

Blackwell, A. L., & Green, A. L. (2008). The abstract is "an enemy". Alternative perspectives to computational thinking. In *PPIG 08: Proceedings of the 20th workshop of the Psychology of Programming Interest Group*. 10–12 Sept 2008, Lancaster, UK.

Blackwell, A. F., & Morrison, C. (2010). A logical mind, not a programming mind: Psychology of a professional end user. In *PPIG 2010: Proceedings of the 22nd annual workshop of the psychology of programming interest group, September 19–22, 2010. Madrid, Spain*(pp. 175–184). University of Lancaster.

Bowers, J. M. (1991). The Janus faces of design: Some critical questions for CSCW. In J. M. Bowers & S. D. Benford (Eds.), *Studies in computer supported cooperative work* (pp. 333–350). Amsterdam, The Netherlands: North-Holland Publishing Co. Retrieved from dl.acm.org/citation.cfm?id = 117730.117753

Bramming, P., Hansen, B. G., Bojesen, A., & Olesen, K. G. (2012). (Im)perfect pictures: Snaplogs in performativity research. *Qualitative Research in Organizations and Management: An International Journal, 7*(1), 54–71. doi:10.1108/17465641211223465.

Brand, S. (1995). *How buildings learn: What happens after they're built*. New York: Penguin Books.

Bratteteig, T., Morrison, A., & Wagner, I. (2010). Research practices in digital design. In *Exploring digital design: Multi-disciplinary design practices* (pp. 17–54). Berlin: Springer.

Bringay, S., Barry, C., & Charlet, J. (2006). Annotations: A functionality to support cooperation, coordination and awareness in the electronic medical record. In *COOP'06: Proceedings of the 7th international conference on the design of cooperative systems*, France, Provence.

Bryant, A. (2000). It's engineering Jim… but not as we know it: Software engineering; solution to the software crisis, or part of the problem? In *ICSE'00: Proceedings of the 22nd international conference on software engineering* (pp. 78–87). New York: ACM. doi:10.1145/337180.337191.

Bucciarelli, L. (2003). Designing and learning: A disjunction in contexts. *Design Studies*, *24*(3), 295–311. doi:10.1016/S0142-694X(02)00057-1

Buescher, M., Gill, S., Mogensen, P., & Shapiro, D. (2001). Landscapes of practice: Bricolage as a method for situated design. *Computer Supported Cooperative Work (CSCW), 10*(1), 1–28. doi:10.1023/A:1011293210539.

Butler, J. (1993). *Bodies that matter: On the discursive limits of sex*. New York: Routledge.

Cabitza, F. (2011). "Remain Faithful to the Earth!": Reporting experiences of artifact-centered design in healthcare. *Computer Supported Cooperative Work (CSCW), 20*(4), 231–263. doi:10.1007/s10606-011-9143-1.

Cabitza, F. (2014). De-designing the IT artifact. Drafting small narratives for the coming of the Socio-Technical Artifact. In *ItAIS 2014, Proceedings of the 11th conference of the Italian chapter of AIS - Digital innovation and inclusive knowledge in times of change, track on design and re-design of socio-technical systems*, November 21–22, 2014, Genova, Italy.

Cabitza, F., & Gesso, I. (2011). Web of active documents: An architecture for flexible electronic patient records. In A. Fred, J. Filipe, & H. Gamboa (Eds.), *Biomedical engineering systems and technologies* (Vol. 127, pp. 44–56). IADIS, Berlin/Heidelberg: Springer.

Cabitza, F., & Gesso, I. (2012). Rule-based programming as easy as a child's play. A user study on active documents. In *IHCI'12: IADIS international conference interfaces and human computer interaction 2012 Lisbon, Portugal, 21–23 July 2012* (pp. 73–80). IADIS Press (online).

Cabitza, F., & Simone, C. (2010). WOAD: A framework to enable the end-user development of coordination oriented functionalities. *Journal of Organizational and End User Computing (JOEUC), 22*(2), 1–20. doi:10.4018/joeuc.2010101905.

Cabitza, F., & Simone, C. (2012a). "Whatever Works": Making sense of information quality on information system artifacts. In G. Viscusi, G. M. Campagnolo, & Y. Curzi (Eds.), *Phenomenology, organizational politics, and IT design: The social study of information systems* (pp. 79–110). IGI Global. Retrieved from 10.4018/978-1-4666-0303-5.ch006

Cabitza, F., & Simone, C. (2012b). Affording mechanisms: An integrated view of coordination and knowledge management. *Computer Supported Cooperative Work (CSCW), 21*(2), 227–260. doi:10.1007/s10606-011-9153-z.

Cabitza, F., & Simone, C. (2012c). Design Ltd.: Renovated Myths for the Development of Socially Embedded Technologies, arXiv:1211.5577v3 [cs.HC].

Cabitza, F., & Simone, C. (2013a). Computational coordination mechanisms: A tale of a struggle for flexibility. *Computer Supported Cooperative Work (CSCW), 22*(4–6), 475–529. doi:10.1007/s10606-013-9187-5.

Cabitza, F., & Simone, C. (2013b). "Drops hollowing the Stone". Workarounds as resources for better task-artifact fit. In L. Ciolfi, M. A. Grasso, & O. W. Bertelsen (Eds.), *ECSCW 2013: Proceedings of the 13th European conference on computer supported cooperative work. 21–25 September 2013, Paphos, Cyprus* (pp. 103–122). Berlin: Springer.

Cabitza, F., Simone, C., & Sarini, M. (2009). Leveraging coordinative conventions to promote collaboration awareness. *Computer Supported Cooperative Work (CSCW), 18*(4), 301–330.

Cabitza, F., Corna, S., Gesso, I., & Simone, C. (2011a). WOAD, a platform to deploy flexible EPRs in full control of end-users. In A. Blandford, G. De Pietro, L. Gallo, A. Gimblett, P. Oladimeji, & H. Thimbleby (Eds.), *EICS4Med 2011: Proceedings of the 1st international workshop on engineering interactive computing systems for medicine and health care, co-located with the ACM SIGCHI symposium on Engineering Interactive Computing Systems (EICS 2011) Pisa, Italy, June 13, 2011* (Vol. 727, pp. 7–12). CEUR-WS.org. New York: ACM Press.

Cabitza, F., Gesso, I., & Corna, S. (2011b). Tailorable flexibility: Making end-users autonomous in the design of active interfaces. In K. Blashki (Ed.), *MCCSIS 2011: IADIS multi conference on computer science and information systems, Rome, Italy, July 20–26, 2011* (pp. 20–26). IADIS.

Cabitza, F., Gesso, I., & Simone, C. (2012a). Providing end-users with a visual editor to make their electronic documents active. In *VL/HCC 2012: Proceedings of short papers of the IEEE symposium on visual languages and human-centric computing, September 30–October 4, 2012, Innsbruck, Austria* (pp. 171–174). Innsbruck: IEEE Computer Press. doi:10.1109/VLHCC.2012.6344509.

Cabitza, F., Simone, C., & Locatelli, M. P. (2012b). Supporting artifact-mediated discourses through a recursive annotation tool. In *GROUP'12: Proceedings of the 17th ACM* *international conference on Supporting group work* (pp. 253–262). New York: ACM. doi:10.1145/2389176.2389215.

Cabitza, F., Colombo, G., & Simone, C. (2013). Leveraging underspecification in knowledge artifacts to foster collaborative activities in professional communities. *International Journal of Human - Computer Studies, 71*(1), 24–45. doi:10.1016/j.ijhcs.2012.02.005.

Cabitza, F., Fogli, D., & Piccinno, A. (2014a). "Each to His Own": Distinguishing tasks, roles and artifacts in EUD practices. In L. Caporarello, B. Di Martino, & M. Martinez (Eds.), *Smart organizations and smart artifacts – Fostering interaction between people, technologies and processes* (Vol. 7, pp. 193–206). Berlin/Heidelberg: Springer.

Cabitza, F., Fogli, D., & Piccinno, A. (2014b). Fostering participation and co-evolution in sentient multimedia systems. *Journal of Visual Languages and Computing, 25*(6), 684–694.

Cadiz, J. J., Gupta, A., & Grudin, J. (2000). Using web annotations for asynchronous collaboration around documents. In *CSCW 2000: Proceedings of the 2000 ACM conference on Computer supported cooperative work* (pp. 309–318). New York: ACM Press. doi:10.1145/358916.359002.

Campbell-Kelly, M., & Aspray, W. (Eds.). (2004). *Computer: A history of the information machine*. Boulder, CO: Westview Press.

Carroll, J. (2004). Completing design in use: Closing the appropriation cycle. In *European conference on information systems* (paper 44). Association for Information Systems. http://aisel.aisnet.org/ecis2004/44

Carroll, J. M., Kellogg, W. A., & Rosson, M. B. (1991). The task-artifact cycle. In J. M. Carroll (Ed.), *Designing interaction: Psychology at the human-computer interface* (pp. 74–102). New York: Cambridge University Press.

Carstensen, P. H., Sorensen, C., & Borstrom, H. (1995). Two is fine, four is a mess: Reducing complexity of articulation work in manufacturing. In *COOP'95, proceedings of the international workshop on the design of cooperative systems* (pp. 314–333). Sophia Antipolis, FR: INRIA.

Carr, N. G. (2004). Does IT matter?: Information technology and the corrosion of competitive advantage. Boston: Harvard Business School Press.

Chen, W., & Akay, M. (2011). Developing EMRs in developing countries. *IEEE Transactions on Information Technology in Biomedicine, 15*(1), 62–65. doi:10.1109/TITB.2010.2091509.

Christensen, L. R. (2014). Practices of stigmergy in the building process. *Computer Supported Cooperative Work (CSCW), 23*(1), 1–19. doi:10.1007/s10606-012-9181-3.

Ciborra, C. U. (1992). From thinking to tinkering. *Information Society, 8*, 297–309.

Ciborra, C. (2002). *The labyrinths of Information challenging the wisdom of systems*. Oxford/New York: Oxford University Press.

Ciborra, C. (2006). The mind or the heart? It depends on the (definition of) situation. *Journal of Information Technology, 21*, 129–139.

Clancey, W. J. (1997). *Situated cognition: On human knowledge and computer representations*. Cambridge/New York: Cambridge University Press.

Cooke-Davies, T. (2002). The "real" success factors on projects. *International Journal of Project Management, 20*(3), 185–190. doi:10.1016/S0263-7863(01)00067-9.

Crabtree, A., Rodden, T., & Bb, N. N. (2001). *Wild sociology: Ethnography and design*. Lancaster: Department of Sociology for the Degree of Doctor of Philosophy, Lancaster University.

D'Adderio, L. (2008). The performativity of routines: Theorising the influence of artefacts and distributed agencies on routines dynamics. *Research Policy, 37*, 769–789. doi:10.1017/S174413741000024X.

Danholt, P. (2005). Prototypes as performative. In *CC'05: Proceedings of the 4th decennial conference on Critical computing, between sense and sensibility* (p. 1). New York: ACM Press. doi:10.1145/1094562.1094564.

De Michelis, G. (2003). The Swiss Pattada. *Interactions, 10*(3), 44–53. doi:10.1145/769759.769760.

De Michelis, G., Loregian, M., Moderini, C., Marti, P., Colombo, C., Bannon, L., Storni, C., & Susani, M. (2009a). Designing interaction for next generation personal computing. In T. Gross, J. Gulliksen, P. Kotzé, L. Oestreicher, P. Palanque, R. O. Prates, & M. Winckler (Eds.), *Human-computer interaction – INTERACT 2009* (Vol. 5727, pp. 926–927). Berlin/Heidelberg: Springer.

De Michelis, G., Loregian, M., & Moderini, C. (2009b). Itsme: Interaction design innovating workstations. *Knowledge, Technology & Policy, 22*(1), 71–78.

De Souza, C. S., & Leitao, C. F. (2009). Semiotic engineering methods for scientific research in HCI. *Synthesis Lectures on Human-Centered Informatics, 2*(1), 1–122. doi:10.2200/S00173ED1V01Y200901HCI002.

DeLone, W. H., & McLean, E. R. (1992). Information systems success: The quest for the dependent variable. *Information Systems Research, 3*(1), 60–95. doi:10.1287/isre.3.1.60.

Dirksmeier, P., & Helbrecht, I. (2008). Time, non-representational theory and the 'Performative Turn' – Towards a new methodology in qualitative social research. *Forum Qualitative Social Research*, *9*(2), Art. 55.

Dix, A. (2007). Designing for appropriation. In *Proceedings of the 21st British HCI Group annual conference on people and computers: HCI... but not as we know it* (pp. 27–30). Swinton: British Computer Society. Retrieved from dl.acm.org/citation.cfm?id = 1531407.1531415

Doerner, C., Draxler, S., Pipek, V., & Wulf, V. (2009). End users at the bazaar: Designing next-generation enterprise- resource-planning systems. *IEEE Software, 26*(5), 45–51.

Dourish, P. (1999). Evolution in the adoption and use of collaborative technologies. In *Proceedings of the ECSCW'99 workshop on evolving use of groupware; 1999 September 16; Copenhagen, Denmark.*

Dourish, P. (2001). *Where the action is: The foundations of embodied interaction*. Cambridge, MA: MIT Press.

Dourish, P. (2004). What we talk about when we talk about context. *Personal and Ubiquitous Computing, 8*(1), 19–30.

Dourish, P., Edwards, W. K., LaMarca, A., Lamping, J., Petersen, K., Salisbury, M., Terry, D. B., & Thornton, J. (2000). Extending document management systems with user-specific active properties. *ACM Transactions on Information Systems, 18*(2), 140–170. doi:10.1145/348751.348758.

Fischer, G., & Giaccardi, E. (2006). Meta-design: A framework for the future of end-user development. In H. Lieberman (Ed.), *End user development – Empowering people to flexibly employ advanced information and communication technology* (pp. 427–457). Dordrecht: Kluwer Academic Publishers.

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:10.1145/1015864.1015884.

Fitzpatrick, G., & Ellingsen, G. (2012). A review of 25 years of CSCW research in healthcare: Contributions, challenges and future agendas. *Computer Supported Cooperative Work (CSCW).* doi:10.1007/s10606-012-9168-0

Flores, F., Graves, M., Hartfield, B., & Winograd, T. (1988). Computer systems and the design of organizational interaction. *ACM Transactions on Information Systems, 6*(2), 153–172. doi:10.1145/45941.45943.

Glendinning, S. (1998). *On being with others Heidegger, Derrida, Wittgenstein*. London/New York: Routledge.

Goodhue, D. L., & Thompson, R. L. (1995). Task-technology fit and individual performance. *MIS Quarterly, 19*, 213–236.

Haigh, T. (2010). *Crisis what crisis? Reconsidering the software crisis of the 1960s and the origins of software engineering*. Milwaukee: School of Information Studies, University of Wisconsin.

Handel, M. J., & Poltrock, S. (2011). Working around official applications: Experiences from a large engineering project. In *CSCW'11: Proceedings of the ACM 2011 conference on computer supported cooperative work* (pp. 309–312). ACM. doi:10.1145/1958824.1958870.

Hanseth, O., & Ciborra, C. (2007). *Risk, complexity and ICT*. Cheltenham/Northampton: E. Elgar.

Harel, D. (2008). Can programming be liberated, period? *Computer, 41*(1), 28–37. doi:10.1109/MC.2008.10.

Harel, D., & Marelly, R. (2003). *Come, let's play: Scenario-based programming using LSCs and the play-engine*. Berlin/New York: Springer.

Harris, J., & Henderson, A. (1999). A better mythology for system design. In *CHI'99: Proceedings of the SIGCHI conference on human factors in computing systems: The CHI is the limit* (pp. 88–95). New York: ACM Press. doi:10.1145/302979.303003.

Hartswood, M., Procter, R., Rouncefield, M., & Slack, R. (2000). Being there and doing IT in the workplace: A case study of a co-development approach in healthcare. In T. Cherkasky (Ed.), *Proceedings of the CPSR/IFIP WG 9.1 participatory design conference* (pp. 96–105). New York: ACM Press.

Heeks, R. (2006). Health information systems: Failure, success and improvisation. *International Journal of Medical Informatics, 75*, 125–137.

Hirschheim, R., & Klein, H. K. (1989). Four paradigms of information systems development. *Communications of the ACM, 32*(10), 1199–1216. doi:10.1145/67933.67937.

Hollnagel, E., Nemeth, C. P., & Dekker, S. (2008). *Resilience engineering perspectives*. Aldershot/Burlington: Ashgate.

Hug, D. (2010). Performativity in design and evaluation of sounding interactive commodities. In *AM'10: Proceedings of the 5th audio mostly conference: A conference on interaction with sound* (pp. 1–8). New York: ACM Press. doi:10.1145/1859799.1859806.

Illich, I. (1973). *Tools for conviviality*. New York: Harper & Row.

Illich, I. (1977). *Disabling professions*. London: Marion Boyars.

Jacucci, C., Jacucci, G., Wagner, I., & Psik, T. (2005). A manifesto for the performative development of ubiquitous media. In *CC'05: Proceedings of the 4th decennial conference on Critical computing: Between sense and sensibility* (pp. 19–28). New York: ACM Press. doi:10.1145/1094562.1094566.

Jensen, T. E. (2002). *Performing social work, competence, orderings, spaces and objects* (PhD), University of Copenhagen, Department of Psychology.

Jensen, C. B. (2005). An experiment in performative history: The electronic patient record as a future-generating device. *Social Studies of Science, 25*(2), 241–267.

Jensen, C. B. (2008). CSCW design reconceptualized through science studies. In *Cognition, communication and interaction: Transdisciplinary perspectives on interactive technology* (pp. 132–148). London: Springer.

Johannessen, L. K., Gammon, D., & Ellingsen, G. (2012). Users as designers of information infrastructures and the role of generativity. *AIS Transactions on Human-Computer Interaction, 4*(2), 72–91.

Johnson, S. (1759). *The history of Rasselas. The Prince of Abyssinia*. London: R. and J. Dodsley, W. Johnston.

Kaplan, B., & Harris-Salamone, K. D. (2009). Health IT success and failure: Recommendations from literature and an AMIA workshop. *Journal of the American Medical Informatics Association, 16*(3), 291–299. doi:10.1197/jamia.M2997.

Khosravi, K., & Gueheneuc, Y. (2004). *A quality model for design patterns*. Montreal: Universite de Montreal.

Kling, R., Rosenbaum, H., & Sawyer, S. (2005). *Understanding and communicating social Informatics: A framework for studying and teaching the human contexts of information and communication technologies*. Medford, NJ.: Information Today, Inc.

Klein, G., & Jiang, J. J. (2001). Seeking consonance in information systems. *Journal of Systems and Software, 56*(2), 195–202. doi:10.1016/S0164-1212(00)00097-2.

Krebs, D., Conrad, A., & Wang, J. (2012). Combining visual block programming and graph manipulation for clinical alert rule building. In *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts*(pp. 2453–2458). New York: ACM. doi:10.1145/2212776.2223818

Kriplean, T., Toomin, M., Morgan, J., Borming, A., & Ko, A. J. (2012). Is this what you meant? Promoting listening on the web with reflect. In *CHI 2012: Proceedings of the international conference on human computer interaction, May 5–10, 2012, Austin, Texas, USA*(pp. 1559–1568). New York: ACM Press.

Lanzara, G. F. (1999). Between transient constructs and persistent structures: Designing systems in action. *Journal of Strategic Information Systems, 8*, 331–349.

Latour, B. (1996). On interobjectivity. *Mind, Culture, and Activity, 3*(4), 228–245. doi:10.1207/s15327884mca0304_2.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. Cambridge: Cambridge University Press.

Law, J., & Singleton, V. (2003). *This is not an object.* Centre for Science Studies, Lancaster University, Lancaster. Retrieved from www.comp.lancs.ac.uk/sociology/papers/Law-Singleton-This-is-Not-an-Object.pdf

Levi-Strauss, C. (1966). *The savage mind (La pensee suavage)*. London: Weidenfeld and Nicolson.

Licoppe, C. (2010). The performative turn in science and technology studies. *Journal of Cultural Economy, 3*(2), 181–188. doi:10.1080/17530350.2010.494122.

Lieberman, H., Paternò, F., Klann, M., & Wulf, V. (2006). End-user development: An emerging paradigm. In *End user development* (Vol. 9, pp. 1–8). Amsterdam: Kluwer Academic Publishers.

Locatelli, M. P., & Simone, C. (2010). A community based metaphor supporting EUD within communities. In G. Santucci (Ed.), *Proceedings of the international conference on advanced visual interfaces, AVI 2010, Roma, Italy, May 26–28, 2010* (p. 406). New York: ACM Press.

Louridas, P. (1999). Design as bricolage: Anthropology meets design thinking. *Design Studies, 20*(6), 517–535. doi:10.1016/S0142-694X(98)00044-1.

Luff, P., Heath, C., & Greatbatch, D. (1992). Tasks-in-interaction: Paper and screen based documentation in collaborative activity. In *CSCW'92: Proceedings of the 1992 ACM conference on Computer-supported cooperative work* (pp. 163–170). New York: ACM Press. doi:10.1145/143457.143475.

Lyotard, J.-F. (1986). *The postmodern condition: A report on knowledge*. Manchester: Manchester University Press.

Lyytinen, K., & Robey, D. (1999). Learning failure in information systems development. *Information Systems Journal, 9*(2), 85–101. doi:10.1046/j.1365-2575.1999.00051.x.

Maguire, S., & McKelvey, B. (1999). Complexity and management: Moving from fad to firm foundations. *Emergence, 1*, 19–61.

Mamlin, B. W., Biondich, P. G., Wolfe, B. A., Fraser, H., Jazayeri, D., Allen, C., Miranda, J., Tierney, W. M. (2006). Cooking up an open source EMR for developing countries: OpenMRS - A recipe for successful collaboration. *AMIA Annual Symposium Proceedings, 2006*, 529. *Managing the human side of information technology: Challenges and solutions*. (2002). Hershey, PA: Idea Group Pub.

Mansfield, J. (2010). *The nature of change or the law of unintended consequences*. London: Imperial College Press.

Mark, G. (2002). Conventions and commitments in distributed CSCW Groups. *Computer Supported Cooperative Work (CSCW), 11*(3), 349–387.

Mark, G., & Semaan, B. (2008). Resilience in collaboration. In *CSCW'08: Proceedings of the 2008 ACM conference on Computer supported cooperative work* (pp. 137–146). New York: ACM Press. doi:10.1145/1460563.1460585.

Mark, G., Gonzalez, V. M., Sarini, M., & Simone, C. (2002). Reconciling different perspectives: An experiment on technology support for articulation. In *COOP'02: Proceedings of the international conference on the design of cooperative systems*. Saint Raphael (FR), 4–7 June (pp. 23–37). Amsterdam: IOS Press.

Martin, D., & Sommerville, I. (2004). Patterns of cooperative interaction: Linking ethnomethodology and design. *ACM Transactions on Computer-Human Interaction, 11*(1), 59–89. doi:10.1145/972648.972651.

Maturana, H. R., & Varela, F. J. (1992). *The tree of knowledge: The biological roots of human understanding*. Boston: Shambhala Publications.

Miller, R. G. (1995). Improving community service: Strategic cooperation through communication. *Communicating Organizational Change: A Management Perspective*, 65.

Mørch, A. I., Stevens, G., Won, M., Klann, M., Dittrich, Y., & Wulf, V. (2004). Component-based technologies for end-user development. *Communications of the ACM, 47*(9), 59–62.

Morrison, C., & Blackwell, A. (2009). Observing end-user customisation of electronic patient records. In V. Pipek, M. Rosson, B. de Ruyter, & V. Wulf (Eds.), *IS-EUD'09: Proceedings of the 2nd international symposium on end-user development* (Vol. 5435, pp. 275–284). Berlin/Heidelberg: Springer. doi:10.1007/978-3-642-00427-8_16.

Morrison, A., Westvang, E., & Skogsrud, S. S. (2010). Whisperings in the undergrowth: Communication design, online social networking and discursive performativity. In I. Wagner, T. Bratteteig, & D. Stuedahl (Eds.), *Exploring digital design: Multi-disciplinary design practices* (pp. 221–259). London: Springer.

Morrison, C., Fitzpatrick, G., & Blackwell, A. (2011). Multi-disciplinary collaboration during ward rounds: Embodied aspects of electronic medical record usage. *International Journal of Medical Informatics, 80*(8), e96–e111.

Myers, M. D. (1995). Dialectical hermeneutics: A theoretical framework for the implementation of information systems. *Information Systems Journal, 5*(1), 51–70. doi:10.1111/j.1365-2575.1995.tb00089.x.

Nandhakumar, J., & Avison, D. E. (1999). The fiction of methodological development: A field study of information systems development. *Information Technology & People, 12*(2), 176–191. doi:10.1108/09593849910267224.

Nemeth, C. (2003). *The master schedule how cognitive artifacts affect distributed cognition in acute care*. Cognitive Technologies Laboratory.

Niazkhani, Z., Pirnejad, H., van der Sijs, H., & Aarts, J. (2011). Evaluating the medication process in the context of CPOE use: The significance of working around the system. *International Journal of Medical Informatics, 80*(7), 490–506.

O'Neill, J. E. (1992). *The evolution of interactive computing through time-sharing and networking*. Ph.D. dissertation, University of Minnesota, USA.

Orlikowski, W. J. (1992a). The duality of technology: Rethinking the concept of technology in organizations. *Organization Science, 3*(3), 398–427. doi:10.1287/orsc.3.3.398.

Orlikowski, W. J. (1992b). Learning from notes: Organizational issues in groupware implementation. In *CSCW'92: Proceedings of the 1992 ACM conference on computer-supported cooperative work* (pp. 362–369). New York: ACM. doi:10.1145/143457.143549.

Orlikowski, W. J. (1996). Improvising organizational transformation over time: A situated change perspective. *Information Systems Research, 7*(1), 63–92. doi:10.1287/isre.7.1.63.

Orlikowski, W. J. (2000). Using technology and constituting structures: A practice lens for studying technology in organizations. *Organization Science, 11*(4), 404–428.

Orlikowski, W. J. (2006). Material knowing: The scaffolding of human knowledgeability. *European Journal of Information Systems, 15*(5), 460–466.

Orlikowski, W. J. (2007). Sociomaterial practices: Exploring technology at work. *Organization Studies, 28*(9), 1435–1448. doi:10.1177/0170840607081138.

Oudshoorn, N., & Pinch, T. (Eds.). (2005). *How users matter: The co-construction of users and technology*. Cambridge, MA/London: MIT Press.

Paley, J. (2007). Complex adaptive systems and nursing. *Nursing Inquiry, 14*(3), 233–242. doi:10.1111/j.1440-1800.2007.00359.x.

Paley, J., & Eva, G. (2011). Complexity theory as an approach to explanation in healthcare: A critical discussion. *International Journal of Nursing Studies, 48*(2), 269–279. doi:10.1016/j.ijnurstu.2010.09.012.

Pan, G., Hackney, R., & Pan, S. L. (2008). Information systems implementation failure: Insights from prism. *International Journal of Information Management, 28*(4), 259–269. doi:10.1016/j.ijinfomgt.2007.07.001.

Papazoglou, M. P., Traverso, P., Dustdar, S., & Leymann, F. (2007). Service-oriented computing: State of the art and research challenges. *Computer, 40*, 38–45.

Perrow, C. (1999). *Normal accidents: Living with high risk technologies*. Princeton, NJ: Princeton University Press.

Pesic, M., Schonenberg, M. H., Sidorova, N., & Van Der Aalst, W. M. P. (2007). Constraint-based workflow models: Change made easy. In *OTM'07: Proceedings of the 2007 OTM Confederated international conference on On the move to meaningful internet systems: CoopIS, DOA, ODBASE, GADA, and IS - Volume Part I* (pp. 77–94). Berlin/Heidelberg: Springer.

Peters, I. (2006). Against folksonomies - Indexing blogs and podcasts for corporate knowledge management. In *Proceedings of online information, London, UK* (pp. 93–97). London: Learned Information Europe.

Pickering, A. (1995). *The mangle of practice: Time, agency and science*. University of Chicago Press.

Pickering, A. (2008). Beyond design: Cybernetics, biological computers and hylozoism. *Synthese, 168*(3), 469–491. doi:10.1007/s11229-008-9446-z.

Pipek, V., & Wulf, V. (2009). Infrastructuring: Toward an integrated perspective on the design and use of information technology. *Journal of the Association for Information Systems, 10*(5), 1.

Riemer, K., & Johnston, R. B. (2012). What is IT in use and why does it matter for IS design? In *Proceedings of the IT Artefact Design & Workpractice Intervention, A Pre-ECIS and AIS SIG Prag Workshop, June 10, 2012, Barcelona, E*. Forskningsnaetverket VITS. Retrieved from www.vits.org/uploads/IT_Artifact/What_is_IT_and_why_does_it_matter.pdf

Robey, D., & Markus, M. L. (1984). Rituals in information system design. *MIS Quarterly, 8*(1), 5–15.

Robinson, M. (1993). Design for unanticipated use. In *Third European conference on computer-supported cooperative work* (pp. 187–202). Milano: Kluwer Academic Publishers.

Robinson, M., & Bannon, L. (1991). Questioning representations. In *ECSCW'91: Proceedings of the second European conference on computer-supported cooperative work*. Amsterdam, The Netherlands.

Rochlin, G. (1998). *Trapped in the net: The unanticipated consequences of computerization*. Princeton: Princeton University Press.

Rorty, R. (1991). *Objectivity, relativism, and truth*. Cambridge/New York: Cambridge University Press.

Rusk, N., Resnick, M., Berg, R., & Pezzalla-Grandlund, M. (2008). New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology, 17*(1), 59–69.

Schmidt, K. (1999). Of maps and scripts: The status of formal constructs in cooperative work. *Information and Software Technology, 41*(6), 319–329. doi:10.1016/S0950-5849(98)00065-2.

Schmidt, K. (2011a). *Cooperative work and coordinative practices: Contributions to the conceptual foundations of Computer-Supported Cooperative Work (CSCW)*. New York: Springer.

Schmidt, K. (2011b). Dispelling the mythology of computational artifacts. In *Cooperative work and coordinative practices contributions to the conceptual foundations of Computer-Supported Cooperative Work (CSCW)* (pp. 391–413). Berlin: Springer.

Shapiro, D. (2005). Participatory design: The will to succeed. In *CC'05: Proceedings of the 4th decennial conference on critical computing: Between sense and sensibility* (pp. 29–38). New York: ACM Press. doi:10.1145/1094562.1094567.

Shipman, F. M., & Marshall, C. C. (1999). Formality considered harmful: Experiences, emerging themes, and directions on the use of formal representations in interactive systems. *Computer Supported Cooperative Work, 8*(4), 333–352.

Simon, J. (2010). *Knowing together: A social epistemology for socio-technical epistemic systems* (PhD). Universitat Wien.

Simone, C., & Schmidt, K. (1993). *Computational mechanisms of interaction for CSCW, COMIC deliverable D3.1, Esprit basic research project*, Lancaster, U.K.

Star, S. L., & Bowker, G. C. (1999). *Sorting things out: Classification and its consequences*. London: MIT Press.

Star, S. L., & Strauss, A. (1999). Layers of silence, arenas of voice: The ecology of visible and invisible work. *Computer Supported Cooperative Work, 8*, 9–30.

Stevens, G. (2010). *Understanding and designing appropriation infrastructures: Artifacts as boundary objects in the continuous software development*. University of Siegen.

Stevens, G., Quaisser, G., & Klann, M. (2006). Breaking it up: An industrial case study of component-based tailorable software design. In H. Lieberman, F. Paternò, & V. Wulf (Eds.), *End user development* (Vol. 9, pp. 269–294). Springer: Dordrecht. Retrieved from dx.doi.org/10.1007/1-4020-5386-X_13

Suchman, L. A. (1987). *Plans and situated actions: The problem of human-machine communication*. Cambridge: Cambridge University Press.

Suchman, L. (1994). Do categories have politics? *Computer Supported Cooperative Work (CSCW), 2*(3), 177–190.

Suchman, L. A. (2004). *Figuring personhood in sciences of the artificial*. Department of Sociology, Lancaster University, Lancaster. Retrieved from www.comp.lancs.ac.uk/sociology/papers/suchman-figuring-personhood.pdf

Suchman, L. (2006). *Human-machine reconfigurations: Plans and situated actions*. Cambridge: Cambridge University Press.

Suchman, L., Trigg, R., & Blomberg, J. (2002). Working artefacts: Ethnomethods of the prototype. *British Journal of Sociology, 53*(2), 163–179. doi:10.1080/00071310220133287.

Sumner, T., & Stolze, M. (1997). Evolution, not revolution: Participatory Design in the Toolbelt Era. In M. Kyng & L. Mathiassen (Eds.), *Computers and design in context* (pp. 1–26). Cambridge, MA: MIT Press.

Szewczak, E., & Snodgress, C. (Eds.). (2002). Managing the human side of information technology: Challenges and solutions. Hershey, PA: Idea Group Publishing.

Telier, A. (2011). *Design things* (K. Friedman & E. Stolterman, Eds.). Cambridge, MA: The MIT Press.

Thomas, G., & Fernández, W. (2008). Success in IT projects: A matter of definition? *International Journal of Project Management, 26*(7), 733–742. doi:10.1016/j.ijproman.2008.06.003.

Truex, D. P., Baskerville, R., & Klein, H. K. (1999). Growing systems in emergent organizations. *Communications of ACM, 42*(8), 117–123.

Turner, P. (2005). Affordance as context. *Interacting with Computers, 17*(6), 787–800. doi:10.1016/j.intcom.2005.04.003.

Van der Aalst, W. M. P., Pesic, M., & Schonenberg, H. M. (2009). Declarative workflows: Balancing between flexibility and support. *Computer Science - Research and Development, 23*(2), 99–113. doi:10.1007/s00450-009-0057-9.

Van House, N. A. (2009). Collocated photo sharing, story-telling, and the performance of self. *International Journal of Human-Computer Studies, 67*(12), 1073–1086. doi:10.1016/j.ijhcs.2009.09.003.

VV. AA. (2001). *Extreme Chaos*. The Standish Group International, Inc. Retrieved from www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf

Wagner, I., Stuedahl, D., & Bratteteig, T. (2010). *Exploring digital design: Multi-disciplinary design practices*. London: Springer-Verlag London Limited. Retrieved from dx.doi.org/10.1007/978-1-84996-223-0

Warkentin, M., Moore, R. S., Bekkering, E., & Johnston, A. C. (2009). Analysis of systems development project risks: An integrative framework. *ACM SIGMIS Database, 40*(2), 8. doi:10.1145/1531817.1531821.

Weick, K. E. (1993). Organizational redesign as improvisation. In G. P. Huber & W. H. Glick (Eds.), *Organizational change and redesign: Ideas and insights for improving performance* (pp. 346–379). New York: Oxford University Press.

Weick, K. E. (2004). Normal accident theory as frame, link, and provocation. *Organization & Environment, 17*(1), 27–31. doi:10.1177/1086026603262031.

Weinstein, D., & Weinstein, M. (1991). Georg Simmel: Sociological flaneur bricoleur. *Theory, Culture Society, 8*, 151–168.

Wilson, M., & Howcroft, D. (2002). Re-conceptualising failure: Social shaping meets IS research. *European Journal of Information Systems, 11*(4), 236–250. doi:10.1057/palgrave.ejis.3000437.

Winograd, T., & Flores, F. (1986). *Understanding computers and cognition: A new foundation for design*. Reading: Addison Wesley.

Winthereik, B. R., & Vikkelso, S. (2005). ICT and integrated care: Some dilemmas of standardising inter-organisational communication. *Computer Supported Cooperative Work (CSCW), 14*(1), 43–67.

Wittgenstein, L. (1922). *Tractatus Logico-Philosophicus*. Cambridge: Dover Publications.

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.