

Conclusion

This dissertation has presented an account of software work in a city in Brazil that challenges some of the common views about the globalization of work and knowledge. On the one hand, my account challenges the trivialized notions of the “flattening” of the world, pointing out not only that place continues to matter today, but also that globalization can even strengthen some of the asymmetries. On the other hand, it demonstrates the extent and importance of transnational connections in modern work —uneven as such connections may be. The dissertation shows globalization as an active process and as comprised of numerous individual globalization projects. Drawing on contemporary ethnographic data and historical accounts, I show how individuals build alliances in personal pursuits of globalization, stressing that such pursuits must be understood through a combination of a cultural and an economic perspectives. I show how such alliances transform local contexts, first establishing landing strips that enable arrival of additional elements of remote practices, then gradually synchronizing local contexts with remote ones. I also discuss the challenges faced by such alliances, stressing the partial and uneven state of globalization, and the combination of isolation and hyper-connectedness that sometimes emerges in the process.

My work takes the form of a “thick description” of software work in Rio de Janeiro (Geertz 1973), combined with an analysis that relies on a theoretical perspective that I believe can help us make sense of globalization of work and knowledge in other contexts. In pursuit of such “thick description,” I have tried to present not just the observed behaviors and the recorded stories of the developers, but also the meaning, rationale and context of their actions. While such representation is inherently interpretative (as is any representation), I have attempted, to the extent possible, to separate my thoughts from those of the participants, making it clear when we disagree and presenting the different perspectives that I have encountered. (Software developers in Brazil, after all, do not always agree on everything.) This multiplicity of stories and perspectives thus presented does not allow for an easy summary, and I will not attempt it here.

Instead, I focus in this conclusion on the theoretical framework used in the dissertation, the idea of looking at knowledge and work as a *practice*, pointing out the value this perspective brings to analysis of work, knowledge and innovation. I also highlight my specific contributions to the refinement of this perspective: first the smaller ones (corresponding to the discussion in chapter 1.1) and then what I see as the most important one —my attempt to globalize the practice perspective by looking at reproduction and synchronization of practice between places (chapter 1.3). I then discuss the extent to which this perspective may be applicable for understanding other contexts.

The Practice Perspective

In this dissertation I have looked at work, knowledge and innovation through the lens of *practice*. While my reliance on the idea of practice cannot be seen as innovative *per se*, the dissertation fortifies this concept and provides an example of this perspective in action. While the primary contribution of my work lies in my attempt to globalize the idea of practice (as described in the next subsection), it is important to first pause to discuss the value of the concept of practice itself.

In my usage, a practice is a historically developed system of activities involving people and objects, and held together by culture and politico-economic relationships.³⁶⁸ My notion of practice draws on several sources, including, first of all, the fusion of the work studies literature with the literature on communities of practice, and, to a lesser extent, the actor-network theory, the theory of structuration, and politico-economic perspectives on labor. (See chapter 1.1.)

My approach to understanding work as practice stresses looking at work as simultaneously a cultural and an economic phenomenon. In other words, the work of software engineers cannot be understood without considering, on the one hand, the extent to which it is affected by the *culture* of software development, i.e. by the shared dispositions and techniques, acquired through active engagement with communities of practitioners. As we saw throughout the dissertation, what software developers do is affected strongly by what they see as “cool,” “fun,” or “elegant” vs what they see as “boring” or “ugly.” (We saw this most clearly in chapter 2.1, though the same idea has run through all chapters in parts 2 and 3.) On the other hand, it would be wrong to ignore the fact that in most cases such work is done in the context of employment — a politico-economic relationship in which the developers offer their labor in exchange for resources that they can use both for basic sustenance and for the acquisition of objects required for the participation in the cultural side of the practice. (This would include, for example, the acquisition of the latest gadgets, or using money to hire others to assist in a pursuit of a culturally-motivated technological vision.) This idea shows most clearly in chapters 3.1 and 3.4, though it is exemplified throughout the dissertation.

While both perspectives on work are common, they are rarely combined. This dissertation shows the need to do so, by demonstrating how neither of the two is sufficient by itself. A purely cultural perspective (which seems to dominate, for example, the discussion of open source software development), would lead us to politico-economic naiveté and unjustified expectations about the upcoming “flattening” of the world. (It also, can lead us to accepting too quickly the assumptions of the culture we are studying, as we would have no basis from which to critique it. This again is common in the literature on open source.) On the other hand, purely economic approaches to work, and the focus on the control over the labor process, would lead us to put too much stress on the formal organizational systems, such as firms and industries. While such entities do play an important role, I believe it is important to recognize that knowledge and innovation in software often (and increasingly) produced and shared through lateral ties between

368 Cf. with Schatzki’s (1996) definition as “the temporary unfolding and spatially dispersed nexus of doings and sayings” (p. 89).

individual developers, who are often driven by cultural as much as just economic motivations.

The combined perspective means, among other things, looking at how the developers unify the cultural economic sides of practice, a task that is rarely easy. “Why will no one ever pay you to do anything interesting?” asks a recent message to the Lua mailing list. The question is asked in jest —many software developers stress that being paid to do what is interesting is the biggest attraction of software development as a profession. (Needless to say, this means doing what *software developers* find interesting, since the desire to spend long hours “mapping interrupts” is hardly a universal human trait.) It highlights, however, the frequent challenge of making the work bring simultaneously cultural and economic benefits. I tried to show the interaction between such factors in several chapters, starting with Jason’s stories in chapter 2.1.

My approach to practice also stresses its systemic nature. I see practice as a system of interrelated activities, which requires specific components (people, tools, documents) that must be made to work together. Additionally, each practice is itself an element in a larger system of related practices. My approach here draws to some extent on that of Hughes and Abbott (in particular the division of labor between the practices), but is also influenced strongly by the actor network theory (and in particular Latour), especially when it comes to looking at how the elements are made to work together—the process that actor-network theorists call “enrollment.” This aspect of practice was illustrated in chapters 2.2 and 3.4.

I also highlight the role of reflexivity and imagination in modern practice. In other words, practice changes in part because the practitioners reflect on their actions (in part by reading what is written about them) and because they can imagine alternative futures. This aspect is illustrated most clearly in chapters 3.2 and 3.4. To fully appreciate the importance of this aspect of practice, however, we need to consider the processes of globalization, as described below.

Globalizing Practice

The second task undertaken by this dissertation involved extending the practice perspective to account for the phenomena which I believe it was not fully prepared to explain. I attempted in this work to *globalize* the practice perspective by seeking to show how a practice developed in one place takes root in another. This issue is theorized in chapter 1.3 and illustrated throughout the rest of the dissertation, using both historical examples (chapter 2.2) and contemporary qualitative data collected (primarily in part 3).

My discussion of software development in Rio de Janeiro positions the city as a *peripheral site in a widely dispersed but highly centralized world* of software development practice (chapter 2.2), which is dominated by a small number of “Meccas.” Local processes orient themselves towards such Meccas in an attempt to draw on their symbolic power and to bring the local practice closer to the remote standards. I stress the incomplete state of the reproduction process: the practice of software development in Brazil can be seen as a *partial replication* of the American practice of software development. (For most places, most certainly including Rio de Janeiro, this incomplete state is permanent, since the continued change of the

practice in the central sites ensures that the job of reproduction is never finished.) This leads me to highlight the “diasporic” situation of the peripheral practitioners, who engage simultaneously in two cultures: the local mainstream culture and the foreign culture of the practice, shown best by chapter 1.2 by looking at the developers’ use of English and Portuguese.

My discussion of the process of reproduction focused on *disembedding* and *re-embedding* (Giddens 1990) as a key element of this process. A practice, understood as a system, cannot move to a new place all at once. Individual elements of the practice, however, can be detached from the system (“disembedded”), moved and inserted (“re-embedded”) into a new context. Such mobile elements may include material objects (see the story of the UNIVAC’s arrival to Brazil in 1960 in chapter 2.2), people (“the Wallauscheks”), ideas (“the Smith Plan” imported from the United States), and documents. As we saw in parts 2 and 3, people attempting to engage in the practice in a new place must reassemble it from disjoint elements brought from other places, and that such re-embedding is often a non-trivial task.

The same applies in reverse: peripheral participants who want to make a contribution to central practices must thoroughly disembed their innovations, making them mobile. As we saw in the case of Lua (chapter 3.2, 3.3), such disembedding does not involve conversion of contextualized elements into some neutral and context-free medium. Rather, it involves removing them from the local context and linking them to the global context of the practice, which, however, is *local* for those in central sites. Knowledge once shared through Portuguese conversation, for example, takes the form of a global book, written not in Esperanto or Volapük, but in English, the language spoken fluently in California but significantly less so in Rio de Janeiro. The price of such disembedding is borne not only by the peripheral innovators, but also by other peripheral users. Luciano’s struggle with reading *Programming in Lua* —an English book written in Brazil just a few kilometers away from where he works —is indicative of this (see chapter 1.2). It also shows how peripheral participants in many ways bear the burden of maintaining the predominance of central sites. It can also draw new boundaries locally.

Building on the idea of practice as a system, I stress *the cumulative nature of the reproduction process*. The process of reproduction of practice happens over time, as a gradual synchronization of context. At each step, elements are brought together and local work is done to make the context more similar to the central sites, thus laying the “tracks” that Latour stresses must be in place for knowledge to move between places (Latour 1988a). Alternatively, we can think of such efforts as creating landing strips for future elements —enclaves of the practice in the midst of otherwise unconquered territory.³⁶⁹ Once the tracks and landing strips are there, importing additional elements becomes easier. Chapter 2.2 shows us how over a number of decades Brazil’s context was brought closer to that of the central sites of the computing world. Establishment of connections to the Internet, for example, transformed the methods for keeping practices in sync. The different projects described in part 3 all contribute to continued synchronization of context.

369 Morais’s (2006) account of the history of aviation in Brazil stresses the difficulty of bringing aviation to remote places, which can be compared to the “bootstrapping” problems described by Staa (2003, see my discussion in chapter 2.2). Places that had built landing strips were easy to include in the growing aviation networks. Adding a new place, however, presented a “bootstrapping” challenge, requiring the aviators to either look for the closest alternatives (roads, fields) or to arrive to such places by means other than airplane.

The parallel nature of the reproduction process leads to *a complex relation between individual and collective efforts of reproducing foreign practices*. The local practitioners must often make a decision whether to cast their lot with local colleagues or to focus on their individual connections to the remote centers. We saw examples of this both within and between practices. For example, the Brazilian government and other Brazilian users of computers had to decide at several points whether to rely on local makes of computers (in the hope of eventually benefiting from cheaper technology more attuned to local needs) or to focus on acquisition of the better foreign machines. Within the practice, software developers in Rio de Janeiro must decide whether to go with the globally established programming language such as Java, or to dedicate their efforts to supporting a local one. Chapters 3.2–3.4 use the case of Lua to show the challenges this presents for local innovation, which must often succeed abroad before being accepted at home.

Seeing software practice as itself an element of a larger system of practices has led me to stress *the parallel nature of the reproduction process*. As we saw most clearly in chapter 2.2, the reproduction of the foreign software practice cannot be understood in isolation from the parallel efforts of people engaged in different practices, all of them pursuing their own globalization projects. The fact that the centers of many practices coincide simplifies this task tremendously. The work of Alta's engineers, who have mastered technology developed on the west coast of the United States, is made much easier by the fact that their clients seek to emulate business practices originally developed in the same place.

Finally, reflexivity becomes particularly important for understanding the reproduction process. We saw throughout the dissertation that software developers know quite a bit not only about the social context they inhabit (as do most people, argues Giddens 1979), but also about the remote social context. This knowledge of a remote social structure becomes an important structuring tool. Knowledge of how things are done elsewhere can help bring out the same structure locally. It becomes important to look at the sources of such knowledge, and in particular at the developers' use of foreign books and websites not just as a source of technical knowledge but also as a source of ideas for social organization. I have tried to note the use of sources throughout the text.

Foreign structure, however, is not always deemed relevant to the local activities. The developers often know what happens abroad but consider it silly to attempt the same in Brazil. It thus becomes important to pay attention to what outcomes can be *imagined*, and how the dubious nature of such imagination is negotiated. Rodrigo's repeated attempt to draw the distinction between "crazy," "insane" and "maniac" dreams (chapter 3.4) illustrates this point.³⁷⁰ Change often comes from plans that are sufficiently "crazy" to present an ambitious step forward, yet imaginable enough to build a coalition in their pursuit.

370 I do not believe Rodrigo ever meant to properly order those terms and affix each of them a specific degree of craziness. He most certainly understands that those words sound the same to the listener. (The joke would be easier delivered in English by use of the same word with different prosody: "just crazy, not *crazy*.")

Other Places, Other Practices

This dissertation has relied primarily on an observation of a particular practice in a particular place. What can this account tell us about other contexts of work? I start with the question of what this dissertation may tell us about software work in other places. I then briefly discuss whether it may be useful for understanding other kinds of work.

While additional work would be needed to examine the extent to which this perspective would fit the practice of software developers in other places, I expect that for many places such work would discover a substantial fit with the general perspective, if not with the details. This would particularly be the case for other “semi-peripheral” sites such as Rio, places where the software practice has been assembled to a substantial degree but where continued work must be done to keep it up to date with remote standards.

For example, chapter 1.2 would look quite different if it were based on observations of software developers in one of the software capitals of South India. In many parts of India, English is not the language of software, but simply the language of education. It may again be somewhat different if written in Russia, where the local language is in a much stronger position vis-à-vis English than Portuguese is in Brazil, and where programming languages using non-English keywords *have* been developed. In this sense, I believe Brazil represents an intermediate position and a case worth understanding.

The notion that becoming a software developer often has more to do with learning to love the computer than a pursuit of lucrative employment would also not hold for India, where economic considerations appear to be the most important reason for becoming a software developer for many people. Even so, however, the larger perspective taken in this dissertation may well apply to understanding software work in India. While Brazilian software developers learn to love software early on, but may then struggle to find “proper” jobs, software developers in India, for whom exceptional grades in high school often become a ticket to the world’s largest software companies, will likely have to learn to love software after getting their jobs. (If they do not, their status in the global software world will likely remain marginal, as they would be seen by their foreign colleagues as low-cost mercenaries rather than fellow practitioners.) Such differences, fit within the broader perspective presented in my work, though understanding those two different ways of entering the world of software may be a particularly fruitful direction for future work.

This dissertation has focused on a place separated by the centers of the software world by several kinds of distance: the cost of traveling, the differences of languages, national boundaries that limit the movement of both people and things, differences of government policy, national identity, local and national culture. While such different kinds of “distances” quite often coincide, they do not always do so. Looking at places that present specific combinations of those different kinds of distance would help refine the notions of “place” and “peripherality.”

Another question concerns the extent to which the perspective taken in this chapter can be applied to other fields of endeavor. One may wonder if software is unique in the extent to which its practitioners in places such as Rio de Janeiro engage with the global world of practice,

including its global culture. For many other lines of work, the local communities of practice may be all that matters to the individual practitioners. This requires a two-fold answer.

The more abstract aspects of the framework would likely be immediately applicable to a wide range of professions. Even for practices where individual practitioners rarely venture out of their local community, the substantial degree of similarity in practice points to the existence of processes that lead to synchronization of those practices between places, which likely draw on some of the same mechanisms. The perspective taken in this dissertation would in the very least provide a starting point for analysis. For example, I have stressed that the global culture of software development provides a set of “perceptions of judgments,” which include the understanding of software work as interesting and worth pursuing for the sake of intellectual stimulation. This particular way of seeing the practice is most certainly not shared by all other practices. The practitioners of a trade can instead understand their work as a matter of “service,” of “honest work,” as a game, or as form of political action. What is likely to be found across many different practices, however, is the pursuit of a shared understanding of the activity, whatever that understanding may be.

Software may well be nearly unique in the extent to which the use of foreign documents by individual practitioners is important for synchronization of the practice. Software is unique today in the abundance and accessibility of documents describing the practice. It is also unique in the extent to which such documents are *useful*. This likely has to do with the relative immateriality of the software practice. Traditional accounts of science practices, for example, commonly stress the importance of direct access to the material tools and artifacts. Software developers, on the other hand, work with few physical objects apart from their computers. Their work is a disembodied, textual art. Repositories of software code and mailing list (on which code can be shared by simply being pasted into the message) serve as virtual environments in which the objects of work reside and can be observed. Such repositories can be “visited” at little cost, as such visits do not disturb the work that occurs in them.

While there is no equivalent of this for many professions, software represents an example of a *class* of occupations that primarily involve manipulation of digital representations. Software developers, simultaneously help create the technologies that enable work based on digital representations and become pioneers of such work. I expect that a number of such occupations will grow over time, and new technologies will make it possible to increasingly interleave such representations with texts. The reliance on such non-rival representations, may also lead to increased free sharing of elements of practice, enabling non-software equivalents of open source production. As the analysis presented in my work suggests, however, this does not mean that place would cease to matter and may instead add power to the central sites.