

2. Histories and Maps

2.1 Nerds¹⁶⁷

In April of 2007 I sat in an office in one of Rio’s tallest buildings, overlooking Largo da Carioca, talking to “Renato,” a manager in a relatively successful software company, which at the time was experiencing some turmoil. We talked about the company’s challenges, one of which was hiring quickly a large number of “good developers.” (The company’s resources did not allow it to look for “great developers,” Renato explained, so he had to settle for “good” ones.) I asked Renato what “good developers” were like. *I follow Joel Spolsky’s advice*, he replied, referring to a popular US blogger who writes about software development and is widely read by Brazilian software professionals. *I look for people who love to program, who are smart and who get things done. First of all, for people who love to program.*¹⁶⁸ It is easy for him to recognize them, explained Renato, since he loves to code himself. *I simply ask them*, he continued, *‘How did you get into this profession?’ If they say, ‘Oh, I wanted to be an accountant but couldn’t get a job and then I took a course, and...,’ then this is not the right guy. When I hear ‘When I was twelve years old I got my dad’s Amiga...’ — then it’s worth continuing the interview.*

In this chapter I look at how Rio software developers make their early steps towards

167 The word “nerds” is used as a Portuguese word in the title of the chapter. This Portuguese word is pronounced [ˈnɛɾdʒiʃ], which can be approximated as “NEH-jish,” though this would omit the voiceless velar fricative [x], which has no equivalent in English.

168 Spolsky’s essay on interviewing, originally published as a blog post and later included in a book (Spolsky 2000/2004), states two of the principles that Renato mentions: “smart” and “gets things done” (p. 157). While Spolsky discusses the importance of passion in the same essay (p. 160), he does not suggest looking for developers who started to code early or use the phrase “love to program.” Renato is most likely attributing to Spolsky an oft-quoted passage from an essay entitled “Great Hackers,” written by another software personality, Paul Graham:

I know a handful of super-hackers, so I sat down and thought about what they have in common. Their defining quality is probably that they really love to program. Ordinary programmers write code to pay the bills. Great hackers think of it as something they do for fun, and which they’re delighted to find people will pay them for. (Graham 2004/2005, p. 98.)

becoming software developers. I focus my attention on a small number of individuals and try to understand their paths to software development in some detail, while pointing out how their stories are similar or different from those of the other developers whom I interviewed. In the next chapter, I take a macro view of software development and the history of its arrival to Brazil, introducing a broader context for understanding the biographies presented here. In this chapter, however, I leave this larger world mostly undefined, focusing instead more closely on the subjective experiences of the young “nerds” who themselves have a limited understanding of the world they are entering. (We return to individual experiences in part 3, after a macro discussion, this time, however, looking at adult professionals, who often have a much better understanding of where they fit on the map of software.)

In addition to looking at the relations between biography and history, the task and the promise of “sociological imagination” according to C. Wright Mills (1959), the two chapters of this part examine the role of *place*. I look at the future software developers as entering the world of software development *in* a particular place and *from* a particular place, and as joining simultaneously a set of economic relations of production, largely localized, and a global world of practice. While I leave the most focused discussion of geography for the next chapter, I point out in my discussion of individual entry into software the many ways in which the experience of young Brazilian “nerds” reflects their peripheral position and foretells some of the issues that we will discuss in part 3.

In Love with Technology

As Renato’s interviewing strategy indicates, many successful developers start early, and “fall in love” (*ficam apaixonados*) with software before they start practicing it professionally. For them, programming becomes a hobby before it becomes a profession. The practitioners often say that good developers do not *do* technology: they *live* it and *love* it. The choice of software development as a career becomes a matter of finding an arrangement that would allow those who “love” programming to make a living while doing what they enjoy and find interesting. Jobs, consequently, are often judged by the extent to which they support the practice of software development or interfere with it.¹⁶⁹

While “living” the practice is implied in most of the “Chicago school” accounts reviewed

169 In the same essay as quoted in footnote 168, Graham writes:

Great programmers are sometimes said to be indifferent to money. This isn’t quite true. It is true that all they really care about is doing interesting work. But if you make enough money, you get to work on whatever you want, and for that reason hackers *are* attracted by the idea of making really large amounts of money. But as long as they still have to show up for work every day, they care more about what they do there than how much they get paid for it.
(<http://www.paulgraham.com/gh.html>)

While this appears to capture well the experience of many programmers working in the US market, especially the best of them, the Brazilian programmers I interviewed typically displayed more concern about finding work that paid the bills reliably. See also the discussion of Van Maanen & Barley (1984) in chapter 1.1.

in chapter 1.1, the expectation of *loving* it is far from universal. Most practices develop explanations for why the members engage in them, but they often draw on reasons other than “passion.” For example, a professional practice can be understood by the members as a way of making a living without sacrificing freedom of thought (Willis 1977 on manual workers), as service done for the benefit of other people (Orr 1996 on Xerox technicians), or as disciplined and honest work (Lamont 2000 on white working class men).

Renato’s use of passion as a way of identifying “good” developers also shows that even among software developers, not all enter the profession in pursuit of a teenage passion. And of those who do, few later choose jobs based *only* on whether the work aligns with what they love to do. Finding a job that pays, and pays reliably, is typically a major concern, especially for the older developers. Software careers are similar to academic careers in this way. In both cases, the new members are often first drawn to the community of practitioners and their esoteric knowledge. However, those who will continue their engagement with the practice must eventually learn to engage in it in a manner that would allow them to earn a living, freeing them from having to dedicate their time to other kinds of work. To move towards more central and more valued forms of participation often similarly requires learning to engage in the practice in very different forms from the ones that may have originally attracted the novice.

The role of passion may also be specific to particular places. Renato attributes the idea of hiring developers who “love to program” to an American blogger working in New York City, though it was most famously captured by an essayist programmer working in Silicon Valley. In my conversations with software developers in Bangalore, India (including those working for the world’s most prestigious IT companies), I typically heard an explanation that one rarely hears in either Brazil or the United States: “passing for computer science.” “Passing” means getting a high enough score on the national university entrance exams to get into a computer science program. Young people do not *choose* to do software work in India, they are *chosen* for it. Those who get the highest scores on university entrance exams study computer science. Those who score less do other things. The outsourcing economy guarantees computer science graduates such high salaries in comparison to everyone else that few seriously consider not doing computer science when offered a chance. While Indian developers often talk about their love for software as well, they learn to love it later, in college.

In this regard, the situation of Brazilian software developers is more similar to that of their American developers than the Indian ones. While software development provides good career opportunities, it is one of many upper middle class careers in Brazil, and not the best paying. For PUC-Rio’s Department of Informatics, the most prestigious computer science department in Brazil, attracting good applicants for its day time program in “computational engineering” is a challenge at times, says one of the professors. Those who do well on the entrance exams and can afford an expensive day time university face many competing options.

PUC’s cheaper program in “information systems,” taught at night, is more popular, however. It appeals to lower-middle class students who see it as a route to social mobility, though a difficult one. The Brazilian software industry primarily serves domestic clients, who often seek relatively simple systems at low prices. While this may contribute to making the software work less attractive to highly educated Brazilians (who sometimes see themselves as overqualified for

the work they get to do), it also creates many opportunities for less sophisticated software work, at lower wages. Many software companies respond to this by hiring software developers with incomplete college degrees and occasionally with no college experience at all, then relying on a small number of highly educated individuals to manage and mentor them. The lower-tier developers are typically expected to be enrolled and to eventually complete a university program at some university, but can pursue their studies at night. They typically enroll in private universities (though ones much cheaper and far less prestigious than PUC) and often say that they attend them for the sake of the diploma (“a piece of paper”), without expecting to learn.¹⁷⁰

One of my interviewees, Miguel, started his career at the age fourteen as an “office boy” in a software company—an assistant tasked with things like delivering documents to clients, and receiving the minimum salary for the State of Rio de Janeiro, around R\$260 (~US\$140) a month.¹⁷¹ Between the deliveries, Miguel spent time learning to use the computer, and later the basics of web development, relying on conversations with the developers, books and practice (“reading and testing, reading and testing”). He was eventually allowed to take on simple web development tasks needed by the clients. Two years later Miguel joined his current company as an “intern” working on web development and earning R\$300, while attending high school at night. Another six years later, at the time of our interview in 2007, Miguel was considerably more confident in his skills as a developer, was attending a university at night, and was earning between R\$1000 and R\$1500 a month. At age twenty two he was making substantially more than his father, who had not finished high school. Miguel was looking forward to yet higher earnings in the future. Having grown up in a family that he describes as “more towards poor than middle class,” Miguel talked about software development in pragmatic terms—a way to make good living. Some of the other developers whom I interviewed had moved into software in similar ways.

Miguel’s story shows that falling in love with dad’s Amiga in early childhood is not the *only* way to enter the world of software development. (Note, though, that while Miguel enters the world of software through a work environment, he also does this at quite young age.) The path to software that starts as a childhood hobby is an important one, however. Developers who enter software as Miguel did typically stay at the lower rungs of the software industry. (Much of this is of course likely due to the larger class effect.) The more ambitious ones also often look for alternative careers, unless they develop a “passion” for software along the way. At the end of my

170 A small number of my interviewees saw less prestigious private universities as potentially a source of good education, stressing that such universities often higher qualified instructors (including those who teach in the more prestigious public universities during the day). Furthermore, the instructor’s teaching is typically evaluated more stringently in such schools than in public universities, where professors cannot be dismissed. (Stories abound of public university professors who show up for the class just twice in a semester: in the beginning to hand out the syllabus and at the end to hand out the exam.) The main weakness of the private programs thus comes from student’s poor preparation and the programs desire to keep the students enrolled and making progress towards the degree (and paying tuition) regardless of whether they are learning anything. According to one of my interviewees, who attended a private university and later proceeded to a prestigious Master’s program at PUC, a motivated student can take advantage of the qualified personnel, but few students have such motivation. See also the story of Jason’s attendance of a private university below.

171 In 1999, the minimum salary for Rio de Janeiro was around R\$260, with the exchange rate of around R\$1.9 for one US dollar.

interview with Miguel I learned of his plans to apply for a government job unrelated to information technology.

As Renato's words indicate, falling in love with dad's Amiga is also how new members are *supposed* to enter the practice. Workers who have "passion" for software are prized by employers like Renato, who devise various strategies for identifying such developers, though for many the ultimate test is simply the feeling of passion that exudes from such programmers. "You can see it in their eyes," many say. And while some developers may display more passion for software than others, nearly all describe software as something they do out of interest. Those who may have come into it for purely economic reasons are careful to not advertise this fact.

While Renato's preference for programmers who love to program can be viewed as a matter of economic rationality (an attempt to hire people who will work hard without need for supervision and may accept lower pay), it is also a matter of a looking for a cultural fit and signs of central membership in the practice. Understanding software as something that one can enjoy and do "just for fun" is one of the "perceptions" (Becker 1953) that distinguishes members from non-members. People who are passionate about software may be skilled at it because they likely dedicate more of their time to it than those who do not enjoy writing code. However, this passion is often itself a sign of certain experience, of "getting" what software development is all about.

Hangin' Around, Mapping Interrupts

"Zé Luis," who also goes by an English nickname "Jason,"¹⁷² is now in his early thirties and, like the majority of my interviewees, has lived his whole life around Rio de Janeiro. As many other software developers with lower-middle class background, Zé Luis grew up in the suburbs of Rio, in his case in Nova Iguaçu, a large municipality fifty kilometers from Rio, a place he describes as "a peripheral city, in a third world country."

Like many others software developers of his age or younger, Jason's starts his software biography from fairly early childhood:

Jason: I've been playing with computers since I was eight, eight years old. I started working with small computers using Sinclair logic, which in Brazil were commercialized by the name TK 85, TK 82.¹⁷³ Those were really small computers and my dad bought one of them for me, and I developed little games on it, and my cousins, who were the same age as I, played those games, suggested changes, and I would go ahead and implement them. I learned BASIC using the manual of the computer, which came with native support for BASIC. So I learned it there more or less by myself, and got really interested. But I didn't pursue this much further. Actually, I wanted to be a writer, to write fiction. I always had

172 "Zé Luis" is a pseudonym, "Jason" is the actual nickname, used (with Jason's permission) for reasons that will become clear later in the chapter.

173 See the glossary for definitions and additional information on underlined terms.

rather diverse interests, in different areas.

So, it was only years later, when... In the 80s, the education system had a series of problems with the government at that time, for a few years. So there were many strikes and they created gaps of sometimes up to four months during the academic year. [...] During one of those my dad thought it was important to put me in some sort of course so that I wouldn't lose a year without studying. So he put me in a computer [*informática*] course. [...] There in this computer course I was introduced to other technologies: databases, those things. And then eventually got interested in this as a career.¹⁷⁴

Jason's story presents his involvement with computers as happening in two phases. At age eight, his dad got him a computer on which Jason learned to program in BASIC. He did not, however, pursue this interest further at the time. He then came back to programming much later, when he was fourteen. This two-step story is remarkably common, and I believe it reflects the developers' desire to establish the time of their *earliest* experience with computers, since engaging with software in early childhood is one of the ways of demonstrating credentials in a practice that expects passion. (Jason's credentials, however, are put to shame by Célio, quoted later, who starts his story at age *six*.)

Jason got the computer from his father, himself an engineer:

Jason: My dad was an electronic engineer, but he never worked as an electronic engineer. He graduated in Electronic Engineering, but worked giving electronics classes. When I was a kid he worked fixing electronic equipment. He fixed TVs, radios, those kind of things. And my mom was a teacher in an elementary school. To tell the truth, my dad didn't have a fixed job, he worked doing sporadic gigs. And my mom was always a teacher and earned little. [...] We were always short on money, so I don't know how it happened that they could buy this little computer for me.¹⁷⁵

The computer was a substantial expense for the family and was purchased specifically for Jason. While some developers talk about first encountering a computer as a tool actually used by their fathers, Jason's story is more common: a father who does not use a computer himself, buys one for his son, seeing it as something that would be worth learning.

Jason's father's later decision to put his fourteen year old son in an a computer course brought Jason in contact with a social environment where he found friends and mentors, who would help him develop his interest in computers:

Jason: Instead of going to a mall we were *hanging around* [says in English] at this computer course. The instructors of the course were experienced people, experienced professionals, they knew a lot, they were good, and so we would be

174 See appendix C, "Original Interview Quotes" (Jason, June 2007, "TK 85").

175 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Meu pai").

there, picking up tricks and tips from them. People who programmed at a very low level. One guy knew assembler, another one knew C++, another knew C or I don't know.¹⁷⁶ [...] This group of people, we "traded cards" [*trocava figurinhas*], right? We would say: "But how did you manage to do this?" "Ah, I figured out that at such and such interrupt of DOS you can put this thingie and the cursor would then notify you every time that it's... you can intercept the pause at the clock and then you can get the key of the thingie and then you can call this program on top of that one..." Cool ways to do stuff.¹⁷⁷

While access to mentors and peers was quite important as a means of learning about computers, Datacenter also gave Jason access to a milieu in which learning about computers would be understood as *cool*, and where exchange of findings could be integrated with simply "hanging around" with friends—an alternative to going to the mall, as Jason points out. "Trading cards" (roughly equivalent to English "comparing notes," but with a more playful connotation), provided Jason and his peers not only with an opportunity to learn from others, but also with a *reason* for learning new tricks.¹⁷⁸

This social side of Jason's experience should not, however, distract us from the more mundane side of working with software, and the individual effort involved in "pushing horizons" by trial-and-error:

Jason: Those were difficult times, I remember, because finding information was difficult. To figure out how to do something you had to go by trial-and-error. The books were expensive and hard to find, you had to order imported books, so you would go to the bookstore, ask for the catalog, the guy would show you a book. "So there is a book about this?" "Yes, there is." "Then get me one." Then it would take months for the book to arrive, then you would go and buy, and it was crazy expensive. So, a solution normally was to get programs in whatever way possible, someone who had it would make a copy, and you would go and try checking it out and discovering how it worked. Then you would use its resources, and perhaps find someone who had already done something more advanced with this: "Hey, how did you do that?" Then the guy would explain it to you and you would apply it in your program.¹⁷⁹

[...] So it took a lot of time to push our horizons. In return, this was very

176 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Em vez de ir no shopping").

177 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Interceptar no relógio a parada").

178 Two months before our interview I experienced first hand this importance of immediate social milieu. After starting to work on Sputnik (a wiki based on Lua and Kepler, see chapter 3.4) in April 2007, I found this project consuming more and more of my attention, soon becoming quite a bit more interesting than my fieldwork itself. It took me some time to realize the socialization process that I was going through. Time spent working on Sputnik led to ideas that could be discussed over lunch with other Lua developers the following days. Time spent thinking about my fieldnotes, on the other hand, led to insights that had no immediate audience.

179 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Uma época complicada").

thorough [“bem *thorough*, bem minucioso”]. We managed to do things that sometimes surprised the instructors: “Wow, how did you manage *that*?” “Yeah, I had to map all the interrupts there and find out that this one did this, the other one did that. I had to find some way to work around this thing that I couldn’t do.” That happened...¹⁸⁰

Jason stresses the lack of books in those “difficult times” as the reason for having to do by hand the hard work of understanding system’s low level behavior by systematically mapping it out (“mapping interrupts”). This specific problem is rarely mentioned by those who started learning programming later, in the age of Google. One important aspect of software work has remained unchanged, however: now as then, software development requires countless hours of individual work, much of which goes towards understanding why a technical system does what it does and how it could be made to behave differently.

The two sides of software work—the solitary investigation and the social “hanging around”—are inherently linked. Programmers usually understand software work as being, at its best, a process of making discoveries (“cool ways to do stuff”) and sharing them. This sharing helps to expedite individual discovery work and creates an audience for “war stories” (Orr 1996) about the achieved results. To be able to share, however, one must first discover something. And as many programmers point out, the time that one has to spend alone in front of the computer for this turns away all but those who enjoy this process for its own sake. The effort of “mapping interrupts” requires dedication that is seen as obsessive by outsiders, and often by the programmers themselves, who often say that anyone who is not “obsessed” in this way and does not find joy in this painstaking pursuit of obscure knowledge is likely to find this work too frustrating. “In return, this was very *thorough*,” says Jason. Being “thorough” (Jason uses an English word here), is its own reward—a *return* (*compensação*, also “compensation”) for the hours spent with the machine. It is only in the right group of peers, however, that Jason comes to see the “intercepting the pause at the clock” as something “cool,” a legitimate form of “hanging around,” and a reasonable alternative to going to the mall.

After Datacenter, Zé Luis opted for a vocational secondary education (*2º grau técnico*), still in Nova Iguaçu, now thinking of pursuing a career in information technology (*informática*). There he soon found himself ahead of the class and out of touch with his classmates.¹⁸¹

Jason: I finished the primary school in 89, I think. At that point I could program in BASIC, dBase, Clipper, Pascal, a whole bunch of languages. Then I decided to go to a vocational school in IT [*2º grau técnico de informática*], right? Secondary school was a *peace of cake* [says in English]. I passed everything, since I already understood everything that they were offering there, since I already had contact with all of that.

My secondary school was divided into three years: the first was the basics, we

180 See appendix C, “Original Interview Quotes” (Jason, June 2007, “Bem thorough”).

181 Typical graduates of vocational computer schools either end up working as system administrators, often a lower-status occupation, or leave computer industry altogether.

studied all the disciplines: biology, physics, chemistry, and just an introduction to informatics. In the second year the actual technical disciplines started: programming, algorithms, those kind of things. During the first year I discovered a need for automation, for developing a system for the school, and started writing software to keep track of students, for their courses. So that was kind of funny: they had a first year student who wasn't attending technical disciplines yet but was developing a rather advanced system. Which ended up giving me a lot of access to people, the teachers, the labs. I had access to the labs to do this and this gave me the nickname that I use until today professionally.

They called me "Jason," since there was that film "Friday the 13th," about Jason with a mask, etcetera, who would never die. You could shoot him, and then... And I was someone really obsessed with programming, so I would go there and program, spending days there. I studied in the morning that year. [...] And in the afternoon the computer lab was open for me. The access to the lab was restricted, not everyone was allowed to use the computers, but I had access since I was writing this system for the school. So, sometimes I would arrive and start using the computers at noon or so, when the lab opened, and stayed there until ten at night, until they kicked me out. [...]

I remember that once I had an English exam in the morning. I was also good with English, so I did the exam quickly. I finished this English test and left the exam at 8 am, and the labs only opened at one 1 p.m. So I was like: "Man, what am I going to do now to kill time until I can start programming?" So, I went there as if I didn't want anything, to have clear conscience, went there to the lab and it just happened to be open, they were doing maintenance on the computers. So I was like: "Nice, I can go in."

I got there and sat there programming, and the groups were coming and going, coming and going, and I stayed there from eight in the morning until eleven at night without getting up from the chair. And since secondary school is a place where rumors spread naturally, the next day the whole school knew of the boy who had stayed at the computer from eight in the morning until ten at night. So during the next week they started calling me all sorts of names: "zombie," "vampire," "the living dead," "without signs of life." What stuck in the end was "Jason." So, everyone would be like: "Ah, Jason who wouldn't die, who is there at the computer, as always."¹⁸²

Jason's "obsession" with computers was not understood by his peers at the new school (despite the school's technical focus) and marked him as different from fellow students. Jason persevered, however, accepted the nickname, and now wears it with pride, using it as his "professional name" and incorporating it as a part of the name he uses when signing emails: "Zé Luis Jason da Costa." A year later, as Jason got hired as a teaching assistant in his school's computer course, his

182 See appendix C, "Original Interview Quotes" (Jason, June 2007, "O Jason que não morre").

technical knowledge started bringing him certain rewards in terms of social status at school. Another year later, as Jason started an internship at the Naval Research Institute, he met more students like himself and eventually developed a group of friends equally “obsessed” with topics few high school students found interesting.

“In a Place So Far Away”

In addition to individual discovery and the tips picked up from peers and mentors, Jason mentions “imported books” as the source of knowledge about technology. While having to order books from abroad must have highlighted the foreign nature of the practice he was starting to engage with, the power of the remote centers over the local practice was not as apparent to Jason in the late 1980s as it is today.

Jason: We wanted to make applications because at that time there were few applications. There were few things. So, since we understood a bit of programming, we thought that we had what we needed to build those applications and become rich and famous. And it was even more exciting to see that we could build things that were good.

I had a friend at the time, his name was Rogerio, who... Everyone had their own interest and he was the guy fixated on text editors. He wanted to make a text editor because he was tired of the one that existed at the time, called WordStar. [...] So he stated to write a text editor that started to have functionality that was better than WordStar. A 16 year old kid, stuck in a place so far away! And that was cool, this joy...

Yuri: Far away where?

Jason: In Nova Iguaçu, far away from... Even far away from the closest metropolitan center, which was Rio de Janeiro, but also far from the place where commercial software was made, which is there in the United States, there in *Silicon Valley* [says in English], et cetera. So, in a peripheral city in a third world country, the guy managed to make a program that in comparison to the commercial software that was available... you could say: “This software is *good!*” This potential motivated us to study, to learn things.¹⁸³

Jason refers to Nova Iguaçu as “a peripheral city in a third world country”—far even from Rio, not to mention Silicon Valley, the Mecca of the software world.¹⁸⁴ While the physical distance

183 See appendix C, “Original Interview Quotes” (Jason, June 2007, “Num lugar lá bem longe”).

184 Jason’s TK 85 used a CPU by Zilog, a company headquartered in San Jose. WordStar was developed by MicroPro, a company based in California. On the other hand, DOS, C and Sinclair all come from outside the Silicon Valley. Whether or not Jason knew such geographic details at the time, it is clear that “Silicon Valley”

from Silicon Valley to Nova Iguaçu has remained the same since, the meaning of this distance has changed. In 1980s, Jason and his friends were entering the software world in a place that was a lot more isolated. Paradoxically, it was this isolation that helped them dream big. Jason's friend Rogerio wrote his text editor software as an alternative to WordStar. Ironically, by that time (1987–88), WordStar was dramatically losing market share in the US, suffering devastating competition from WordPerfect and Microsoft Word.¹⁸⁵

Rogerio's ambition, however, was no match for Jason's own:

Jason: I remember that at the time I was the guy who liked to do graphics. I wanted to do something that would allow you to run several programs at the same time in different windows. Now you see: I wanted to do this in graphic form, on the DOS screen, but it was very slow, not very good. I wanted to keep trying better solutions, to put smarter video drivers, to copy the data faster. So I arrived at the conclusion that to do this I would have to use the disk and that it would end up being very slow, so I decided that this would not work and gave up. And went to pursue other things. I was quite annoyed when Windows came out a few years later, using of course the disk—which was the idea that I had and discarded as undoable. I thought: “Damn, if I had pursued this, I would have become rich.” [Laughs.] Or not, right? [Long pause.]¹⁸⁶

While Jason remembers Windows coming out “a few years later,” the first version was actually released in 1985, when Jason was ten years old. Jason did not see Windows until 1990. Technical news took time to reach Nova Iguaçu in the 1980s.

This relative isolation made the 1980s a difficult time to do software work, Jason says. It also, however, made it a time of unchecked dreams. As a teenager, Jason thought he would have become rich had he managed to develop a good way of running programs in multiple windows (something that had made Bill Gates wealthy three years earlier). Now he seems to doubt that this would have helped. While this loss of optimism undoubtedly has much to do with growing up, younger developers rarely express the same sense of excitement as those who entered computing in the late 1980s. The Internet has made Brazilian developers simultaneously more and less isolated. While being more connected, Brazilian developers today appear to be more aware of how isolated they are. In 1989, the Silicon Valley was a rather vague idea. It was hard to imagine concretely what it would be like to be there. Today, the developers are lot more exposed to what is happening in the United States. They are thus more aware of being “stuck” (*enfurnado*) in

has iconic meaning, in many ways standing for all of the US software industry (the “et cetera”).

185 Already in 1984 WordPerfect was creating serious competition for WordStar. WordPerfect 4.2, released in 1986, dealt WordStar a blow from which it never recovered. By 1987, WordPerfect had 30% of the US word-processing market, reducing WordStar's share to 16%, with IBM's Displaywrite and Microsoft's Word at 13 and 11 percent respectively (Peterson 1994, p. 115). After an advertising campaign that aimed to deliver a “knockout punch” for WordStar in 1997, WordPerfect Corporation focused its efforts on fighting against Microsoft Word (ibid, p. 122), which was growing in popularity and eventually took over the word processing market.

186 See appendix C, “Original Interview Quotes” (Jason, June 2007, “Tinha ficado rico”).

Brazil.

Perhaps more importantly, the relative scarcity of foreign applications was seemingly creating opportunities for local developers. This situation started changing rapidly in 1990s, as American companies increasingly started to enter the Brazilian market. Local application developers found themselves judged by standards that they were unable to match.

People who start product companies are crazy, says Rodrigo Miranda as we sit down to discuss the history of Nas Nuvens, a company founded by his brother João (see chapter 3.4). *João is crazy in this sense*, he continues. *But he founded Nas Nuvens in 1997. Starting a product company now would be even crazier.* In 1990s the customers knew little of what was happening abroad, explains Rodrigo. Now they compare everything with foreign alternatives. *Then you could say: 'This is a search engine.' And they would say: 'OK.' Now they respond: 'This isn't a search engine. Google is. Is this as good as Google? Does it do the same things?'* As we will see in chapter 3.1, successful companies avoid such competition by building custom software for specific clients, where their location becomes a source of strength vis-à-vis foreign competitors.

The strength of connections to foreign centers varies between Brazilians, and it did so more in the late 1980s than it does today. Unlike Jason, Rodrigo Miranda does not remember any lack of applications. *Hardware was always hard to get in Brazil*, says Rodrigo, *but software was easy. We've even had people from the US ask us to send pirated software from here.* By 1990s, there was “blue box,” he continues, which emitted sounds that tricked the phone network into letting you make free phone calls, even international.¹⁸⁷ Some people he knew, says Rodrigo, used those to connect to BBSs in Sweden for five days at a time. *Yes, a five-day international phone call. You could download a lot of stuff this way.*¹⁸⁸ *One could ask them to download this or that specific thing.* Once one friend had the software, they all had it. “We were used to having new software as soon as one week after it was released,” Rodrigo adds later.

Later there was also a machine on Fundão, says Rodrigo, referring to an island near Rio where one of the universities is located. *It was connected to the Internet and collected larger downloads. Of course its location and its very existence were secret.* I ask him how he knew about it. Rodrigo smiles. *Obviously all the nerds knew each other*, he explains. Living in Rio's upscale Zona Sul and later attending PUC, Rodrigo knew the right people. Fifty kilometers away, in lower middle class Nova Iguaçu, Jason was attending the wrong high school.

In either case, however, the newcomers quickly discovered that they were entering a world centered somewhere far away (even if they were not sure how far) and that success in this world would depend crucially on their ability to build links to those foreign centers. In the very least, they had to obtain access to foreign technology—the hardware and the programs. They had to find the books and learn to read them in English. They also had to learn to build local ties, to

187 In the US “blue box” was often a physical device. In hardware-starved Brazil, it was a program run on a computer, that emitted sound through the computer's speakers. One had to place the ear piece of the phone near the speaker to make the call.

188 “Even considering that connections were typically around 9.6-14.4K baud,” Rodrigo adds in a later email exchange. “The thing is that ‘stuff’ at that time was measured in Kilobytes, not Mega or Gigabytes.” (9.6-14.4K baud is roughly 1/2 the speed of today's “slow” dialup connections, 1/5 of speed of today's “fast” modems, and 1/50 of a typical home broadband connection.)

make construction of global links a collective project.

Being a Nerd

Early in my research I neglected to ask my interviewees about their *earliest* encounters with the world of computers. Luckily, some of them set me straight. At the end of my interview with Célio, a Senior Systems Analyst for a foreign company in Rio, I asked if there were questions he thought I should have asked him.¹⁸⁹ Speaking in short English sentences, separated by measured pauses, Célio responded:

Célio: Maybe when or why I got interested in computer science. This happened when I was six years old and I got an Atari—a video game. And this was my first contact with some electronic device. I was able to compute on something. [...] It was a gift from my father and mother. [...] I saw it on television. I was a kid, just a small kid, so this was just for the fun.¹⁹⁰

Célio's comment took me a bit by surprise, since we *had* talked about his first programming experience, and he told me he started programming in Basic on MSX at age eleven. I thought that he was perhaps referring to the same time:

Yuri: And this was when you were eleven?

Célio: No, I was six. At eleven I already knew that I wanted a computer, and not a video game.

Yuri: So, Atari was just a video game, you couldn't program it?

Célio: No. But in the end video games are programmed. So I decided I wanted to do that for my life. Though I didn't know what "that" was.

In later interviews, many software professionals similarly mentioned video games as one of the first things that got them interested in computers.

Another developer, Mauricio, now in his late twenties, describes the experience of being a "nerd" in high school:

Mauricio: And since I was quite a *nerd*, I spent most of my time in the computer lab.

189 Completing each interview with such a question is recommended by Weiss (1994). In my experience, the interviewees most typically respond to this question a "no" or with a summary of what they have said. Occasionally, however, they use the opportunity to bring up things that did not come up in the interview or to make methodological suggestions, sometimes quite useful.

190 This interview was conducted in English.

Yuri: What does “quite a nerd” mean?

Mauricio: *Geek*. [Pronounces as in English.] I liked that a lot. I wasn’t a very social person.¹⁹¹

He later offers an example:

Mauricio: He [the teacher] would come, give a class, and let people go and the class would go to play soccer. The whole class would leave and we would stay there in the lab. The thing is that Doom came out, so... The big thing to do was to get a mouse and destroy it to make a modem cable. To play Doom against [each other]...

Yuri: Explain...

Mauricio: A cable... for the modem. The serial cable actually, to connect computers. There was no network then. Or, rather, there was, but it didn’t work there. [...] The mouse had the right connector—serial. [...] It was cheaper to get a mouse, break it and make a cable. It got to a point that we had so much practice with this... We would pull it out of the mouse [*picks up an imaginary mouse, rips off its cord and removes the imaginary isolation with his teeth*], connect the wires, attach... It took less than five minutes to make a cable.¹⁹²

Like Jason, Mauricio spent much of his high school time in the computer lab. Unlike Jason, who tells an almost-too-good-to-be-true story of immediate obsession with “mapping the interrupts,” Mauricio talks about a somewhat more prosaic (and seemingly more typical) road: computer games and overall being a nerd.

For many developers, computer games provide the early source of interest in computers. As they spend time playing, many develop an interest in understanding the inner working of the computer, sometimes for pragmatic reasons (finding ways to cheat the game), sometimes out of general fascination (as in Célio’s case). They often learn peripheral skills, from installing software to making a serial cable from a pair of computer mice, as Mauricio’s story indicates. As we saw in chapter 1.2, computer games often also provide future developers with the early motivation for learning English—a crucial skill for their later engagement with software development. Like the world of software development, the world of computer games is centered far away, requiring newcomers to learn how to deal with their peripherality. Perhaps most importantly, however, computer games help them learn the satisfaction that can be found in spending hours in front of the machine. To put it differently, computer games help them learn to enjoy being “nerds.”

While computer games are the most commonly mentioned immediate path to interest in computers, “being a nerd” (*era bem nerd, sou um nerd*) is among the most common *explanations*

191 See appendix C, “Original Interview Quotes” (Mauricio, July 2007, “Como era bem nerd”).

192 See appendix C, “Original Interview Quotes” (Mauricio, July 2007, “Destruir mouse pra fazer o cabo”).

that software developers offer for their their interest in both games and computers. The word “nerd” is written the same way in Portuguese as it is in English, though pronounced differently: “nehji”¹⁹³ in Carioca Portuguese. It has roughly the same meaning as in English, though with a heavier connotation of computer use and often a more derogatory sense. When I ask Mauricio to explain what he means by “being a nerd,” he uses another English word: “geek,” this time pronouncing it just as in English.¹⁹⁴ Most other Brazilian developers who use the word “nerd” similarly avoid defining it. When pushed, they explain it as a matter of interest in computers and not being “social” or offer a humorous definition—“someone who craves machinery instead of carbon-based lifeforms,” jokes one of the “nerds.”

While interest in computers over “carbon-based lifeforms” clearly defines the typical nerd for many nerds (in Brazil as in the US), the word has a broader denotation in both languages, and considering this broader sense will help us understand the role of “being a nerd” in the formation of a software practitioner. One of the versions of Wikipedia’s article “Nerd” defined a “nerd” as “a person who passionately pursues intellectual or esoteric knowledge or pastimes rather than engaging in social life, such as participating in organized sports or other mainstream social activities.”¹⁹⁵ This positive spin on the term (most likely written by a “nerd”) provides a key to understanding the similarities between different ways of “being a nerd” and the consequent transitions between different practices in which the nerds engage. Instead of engaging in “mainstream” activities (such as playing soccer, in Mauricio’s case), nerds find satisfaction in pursuit of “esoteric knowledge.” While they often specialize (becoming “computer nerds” or even “the guy who liked to do graphics”), this pursuit of esoteric knowledge often extends to several domains. Those who find satisfaction in developing software and “mapping interrupts” have often earlier pursued with passion computer games and role-playing games (RPGs). In many ways it is the interest in engaging with *some* system of esoteric knowledge, rather than the choice of a specific system, that defines a nerd.

Nerds are often described (and describe themselves) as “not very social” and are often seen (again by both themselves and the non-nerds) as representing a particular *type* of people: those who *naturally* find satisfaction in pursuits like programming or role-playing games, perhaps being in possession of particular kind of minds, suited for understanding machinery and formal systems, but less for navigating the social environments of the real world. Popular press

193 More precisely, the pronunciation could be represented in IPA as [ˈnɛxɔʒi]. Here [x] represents a voiceless velar fricative, a consonant pronounced as a “harder” version of English “h,” similar to the pronunciation of “ch” in German.

194 The association with computers is somewhat stronger for the Portuguese “nerd” (pronounced *nehji*) than for its English cognate, putting it somewhere in between English “nerd” and “geek.” It appears to be a relatively recent borrowing from English, and its denotation appears to be widening as it spreads. The “native” Portuguese term for “nerd” is “CDF,” an abbreviation for “cu de ferro” (“iron ass”), which refers to people who study too much, with less connotation of computer use. “CDF” is clearly derogatory, while the connotation of “nerd” is somewhat ambiguous. English word “geek” is generally recognized by many developers as a term with a more positive connotation in English than “nerd,” but is rarely used in Portuguese.

195 <http://en.wikipedia.org/w/index.php?title=Nerd&direction=prev&oldid=167702459> (authored on October 27, 2007, last accessed on September 15, 2008). The definition of “nerd” on this page has changed over time and I am using a particular version that I found very apt.

has entertained the idea that many nerds suffer from a mild form of Asperger's syndrome, a psychological disorder that sometimes described as a milder form of autism.¹⁹⁶ While people with mild forms of Asperger's may in fact gravitate to some of the activities that are considered "nerdy," successful engagement in "nerdy" practices typically involves a lot more social interaction than is commonly assumed. As we have seen, Jason's and Rodrigo's pursuit of programming benefited greatly from the right group of peers, while Mauricio learned to make a serial cable from a pair of mice so that he and his friends could connect computers and play Doom together rather than individually.¹⁹⁷ Successful participation in the professional software development similarly requires an acute sense for the social dynamics of software teams and the culture of the software world. This is especially true for open source projects, where the participants must rely on a limited set of clues to communicate with people they barely know.

Becoming a nerd is also fundamentally a social process. "I started with this computer thing around 1981 [actually 1982], but I was doing Texas [Instruments] calculators before, so I already knew something about programming," says Rodrigo Miranda.¹⁹⁸ It was at Computique, Rio's first computer store. "There was one (really one) computer store in Rio at the time, so we used to skip classes at school and go there. [...] It was a [Brazilian clone of] Timex Sinclair ZX-81."¹⁹⁹ The computer store was usually empty, so the staff did not seem to mind a small group of kids staying in front of the computer for hours each day, programming on foot. "They were amused by those kids," says Rodrigo, "Everybody had that IBM look, and they wanted to show to CEOs how that TRS-80 really would revolutionize their companies. So I guess pointing at us and saying, 'See? Even kids can use these new computers!' —that could even help them." Rodrigo got introduced to Z80-based computers by a friend, who recognized in him a fellow "nerd": "A guy in my classroom was into electronics and he was trying to learn how to program in assembler. When he heard that the other nerd used to program a TI58 [a programmable calculator], he assumed I would be able to learn assembler too. So he gave me the Z80 ref man [reference manual]. [...] I simply loved it." Learning to program for Z80 became the next step in Rodrigo's career as a computer nerd, a logical step from the programmable calculator, which Rodrigo had earlier received as a present from his cousin. Even earlier, however, Rodrigo was

196 The popularity of this idea owes in part to Time Magazine's famously diagnosing Bill Gates (in some ways the quintessential "nerd") with Asperger's, and the later reports of off-the-charts autism rates among the children of Silicon Valley. The story appeared in Time (1994) article that quoted side by side two New Yorker articles, one about Bill Gates and another about autism, finding them "strangely and intriguingly similar." The story has since been repeated many times, including by Wired news (Silberman 2001), which also introduced the reports of high autism rates in the Silicon Valley.

197 See Anderegg (2007) for a more extended critique of the link between Asperger's and nerdity.

198 This conversation occurred over IM and I adjusted Rodrigo's IM-style spelling, punctuation and capitalization in this paragraph. See appendix C, "Original Interview Quotes" (Rodrigo, October 2007, "1981").

199 Timex Sinclair ZX-81 was a microcomputer released in 1981 and discontinued in 1983, with a 3MHz processor and 1KB of memory (less than a modern graphical calculator), using a television set for display. It sold in the US for \$100. Rodrigo used TK 82C, a Brazilian clone of ZX-81 produced by Microdigital Eletrônica Ltda. (Jason's TK 85 was a later computer and was, according to Rodrigo, "way better" than TK 82C, "due its vast amount of memory (48K) and the presence of colors." (A typical modern computer has at least 10,000 more memory, of course.) The rapid progress of hardware in the 1980s led many to think, says Rodrigo, that "newer nerds were getting things too easy for them, that they hadn't seen the 'real hard days.'")

“into electronics”: “[assembling] small devices like sirens, LED-based dices,²⁰⁰ door locks, alarms, digital clocks, nothing too fancy, but at 12 this is amazing. [...] people around me were nerdy too, so everybody discovered those things around the same age.” Rodrigo later says that his “LED-based dice” phase happened when he was between eight and ten, adding that assembling electronic devices at such young age “gets you a lot of points with an engineer father.”

As in Jason’s case, two groups of “nerdy” people were important to Rodrigo’s formation as a nerd and later as a software nerd. While it were his peers that helped him develop an interest in programming computers, Rodrigo describes himself as already “a nerd” by the time his friend invited him to the computer store to learn programming a computer. He learned to be a nerd under the influence of his father and other male family members.

Rodrigo’s father, however, was a civil engineer with an interest in electronics, not a software developer and in fact at first disapproved of Rodrigo’s interest in software, not recognizing it as “real” engineering. While being nerds and sons of nerds, most software developers are not sons of software developers, Doom players or RPG fans. Like Rodrigo, they often learn from their fathers and other male relatives some of the key “perceptions and judgments” (Becker 1953, see chapter 1.1), including an appreciation for pursuit of esoteric knowledge and some of the skills required for such pursuit (what they call “learning to learn”). They typically apply such perceptions and judgments to other knowledge systems, however.

This change of focus is often encouraged by the fathers themselves, who see new technological areas emerging and consider them more promising for their sons, while not having the time to pursue those areas themselves. Ultimately, however, while able to bring up their sons as “nerds,” fathers have limited influence over the specific system of knowledge that their sons choose to pursue. They can pass to their sons the general perception of technical knowledge as something that can be explored with satisfaction, but not their choice of the domain. The pursuit of esoteric knowledge must be done in a group of peers, and the specific pursuits of the young nerd are ultimately determined by this peer group. This peer group is needed for its own sake (as any other person, a nerd needs a collective in which he could “belong”), for the help that it can provide in the actual learning process, but also because the existence of a peer group is entailed by the “perceptions and judgments” that make the activity desirable in the first place, since only fellow experts can truly appreciate the advanced levels of esoteric knowledge that the nerd acquires.

Professional Nerds

Being a “nerd” is not a career choice, but a way of life often accepted in childhood. Some systems of knowledge, however, underlie what Giddens (1990) calls “expert systems” — socio-technical systems that are essential for the functioning of the modern society yet are opaque to most of its members. People who master “expert systems” (the “experts”) face good

200 A simple electronic device that shows a pattern of lights corresponding to different patterns that can be obtained with a throw of dice.

opportunities for gainful employment. The system of knowledge related to getting computers to perform various tasks supports one of the Giddensian “expert systems” and appears to be particularly appealing to nerds.

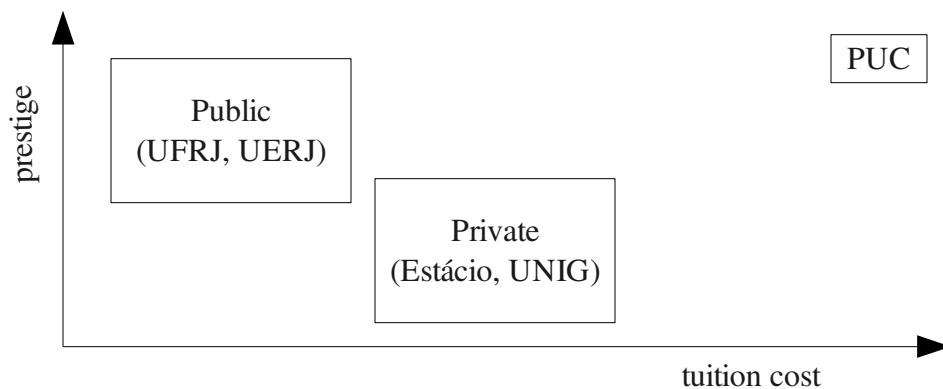
As young nerds grow up, they come to realize (with some nudging by adults) that some systems of knowledge bring more financial opportunities than others and start focusing on them as their future profession, leaving computer games as a hobby. As they move into software development, they bring with them some of the skills they acquired in other “nerdy” pursuits—such as Mauricio’s networking skills perfected for the sake of playing Doom. More importantly, they bring with them a set of Becker’s (1953) “perceptions and judgments” that make it possible to see the software development as enjoyable because of the opportunity to “push horizons” that it offers.

This transition, however, involves more than a choice of one system of knowledge out of a few attractive alternatives. The future software developer must start to engage in an entirely new way with the social world surrounding computer knowledge. As I argued in part 1, a practice such as software development must be understood as simultaneously a culture and a system of economic relationships. A sixteen year old “nerd” who has acquired a good quantity of the culture of software development may nonetheless be quite new to its economic side, and in fact may have to re-learn some of the culture. This transformation typically requires a combination of two factors: experience working in an organization that produces software for commercial use and the acquisition of a broader theoretical base of software development. Most developers acquire this broader base at least in part through college education.

While the developers often stress their ability to learn by themselves and are frequently critical of the undergraduate experience, most of those who spend enough time in college recognize at least some value in the experience beyond the certification demanded by the employers. For those who attend college full-time, the early years often provide a crucial socialization experience, introducing the future software professionals to a much expanded circle of like-minded peers as well as to people who have engaged with software for much longer. Many also stress the importance of the curriculum and the discipline demanded by some of the university programs. “When you go to the university you have a defined curriculum that you have to go through, whether you like it or not,” says Célio, who attended a public university. He illustrates this point with a story of how he learned Java in 1997, at the time when few people at his university even knew of the existence of this new programming language. Célio had to rely on books and the Internet to learn about Java, but credits the university for pushing him to learn by requiring him write a report about a less known programming language. Such learning of course could and does happen in the work place. Few employers, however, want to train their employees from scratch, and most usually require at least a year of college instruction even for those who have learned programming in high school.²⁰¹

Rio residents usually group metropolitan area’s educational institutions into three classes, or rather, two large classes and one institution that stands apart by itself. The classes represent different combinations of prestige and cost, illustrated on Figure N.

201 Gary Becker’s (1962) notion of human capital and Spences’s (1973) discussion of job market signaling provide two useful approaches to thinking about this hiring strategy.



Public universities (*universidades públicas*) are owned and managed by the state, either at the federal level (“federal universities”) or at the state level (“state universities”). Public universities do not charge tuition, but require a high score on an entrance exam (*vestibular*) for the more prestigious courses. (Federal universities, such as UFRJ, are typically more selective than state ones, such as UERJ.) Passing the *vestibular* for a popular course typically requires a prior level of preparation that is not offered by public high schools. To receive public higher education in a popular field one thus often needs to either attend a private high school or private preparation courses (*pre-vestibular*), though the situation has changed somewhat in the recent year with the introduction of quotas for students from public schools. Additionally, public universities typically offer classes during the day, making it more difficult for students to combine work and studies, making such universities most attractive to students who can rely on parents to pay for their living expenses.

Lower middle class students who cannot attend public universities are served by **private universities** (*universidades particulares*), typically run as for-profit institutions, such as Universidade Estácio de Sá. Some, like Universidade Iguazu (UNIG), are set up as non-profit organizations. Such universities charge tuition, but accept students with less preparation and offer night classes, allowing the students to combine studies with full time work.

The third class of educational institutions is **catholic universities** (*universidades católicas*), represented in Rio de Janeiro by a single institution: Pontifical Catholic University of Rio de Janeiro (PUC-Rio or simply “PUC” with the city). Catholic universities are private in the sense that they are not operated by the state, but are not run for profit. They are typically much more expensive than “private” universities (and are thus the most expensive educational options in Brazil) and are comparable in selectivity to the federal universities.

A year after finishing high school, Jason decided to start a university program. Like many of my other interviewees who attended public high schools, he did not see public universities or PUC as an option, since his high school had not prepared him for the entrance exams. Instead, Jason started a night-time program at UNIG. He abandoned the program after six months, concluding that he was not going to learn anything useful in it:

Jason: I only spent three months in that university. [...] At the time I was a very technical guy, not very... I was very much a “bit twiddler” [*escovador de bit*] is

what we called it, right. A really low-level guy. [...] And so I underestimated such knowledge as high-level analysis. I even considered databases trivial: “Meh, you just take the data, put it there, and later take out, put in again and take it out, put in, take out. Nothing special.” So I somewhat underestimated the courses that they had. I thought: “Whatever. They won’t teach me anything new here, so I won’t stay here.” And so I went home to study other things. And I ended up doing a mix of things: studied this graphics thing, studied programming languages, learned C++ at that time, between 1991 and 1994, and other things. [...] At that time there already were decent books, right. It was already 91, 92. It was possible to go to a bookstore. They opened one, actually here, downtown, it was our *playground* [says in English]. “Livraria Ciência Moderna” [“Modern Science Book Store”], here in Edifício Avenida Central...²⁰²

Having up to that point studied largely by himself, Jason had appreciation for certain types of knowledge, which did not include the theoretical and “high level” knowledge that the university program was offering him. While he later realized that such topics are important for his work, at the time a small local university lacked the prestige that might have convinced Jason to put aside his own judgment and take the courses more seriously. The seeming ease of learning everything at home, using foreign books that were becoming increasingly available, further contributed to his doubts about the university.

Jason’s experience with the university illustrates a more general pattern that we will see many times later: peripheral actors often lack trust in each other, which leads them to focus on direct global links rather than on making use of what other local actors can offer. Students assume that local universities (especially the less prestigious ones) cannot teach them anything useful. The university instructors have equally little trust in the students’ ability to learn.

Such quick abandonment of the university program is somewhat atypical, enabled in part by Jason’s success in finding a programming job prior to attending university. In contrast, most software developers I interviewed in Rio de Janeiro had completed at least one or two years of a university program in computer science or *informática* before being able to get any serious software job. The percentage of those who complete their degrees, however, is much lower than it would be in the United States. Those who can get into (and afford) PUC or one of the public schools typically spend their first year or two just studying (and often remember that year fondly as a time of learning), but then look for an “internship” (*estágio*), which often means a relatively permanent job with somewhat reduced hours and additional flexibility, at a reduced salary. They typically finish their program (which typically leads to a raise), but often talk about the last few years of school without enthusiasm. The most important effect of the university program, therefore, is that it helps the future developers obtain an internship.

Those who cannot attend public schools or PUC typically start working in a different (though often related) occupation, for example offering technical support, while simultaneously attending night classes. After a year or two of night classes they can often get an internship, though at a lower pay than PUC students. While they often doubt the quality of instruction in the

202 See appendix C, “Original Interview Quotes” (Jason, June 2007, “Subestimei as cadeiras”).

universities they attend, they typically stay enrolled, knowing that such university enrollment will be seen favorably by future employers and that the eventual “piece of paper” will further expand their employment options. Their lack of trust in the program is thus primarily expressed in lack of attention to the course. After obtaining an internship and eventually more permanent employment, developers who study in private universities at night often slow down their progress substantially, sometimes taking many years to finish the program or giving up altogether.

After working briefly in a few small companies, Jason got a job in Petrobras—Brazil’s semi-public oil company, widely seen in Brazil as one of the few sites of technical innovation. Working for Petrobras had been Jason’s “professional dream,” but turned out more complicated than he expected, as Jason discovered the limits of what he knew.

Jason: It was at Petrobras that I kind of started feeling the crisis of my super-technicism. Because I got there and discovered that the world had changed a little, things were easier to do in the world of 1994, 1995, mid 90s. The technology had become easier, access to information was easier, and I had a lot of technical background in “bit twiddling” but this wasn’t as valued as it was five or six years ago in the late 80s. So I saw a market that needed bank applications, basically information systems, and I didn’t have much theoretical background in this, right. [...] Suddenly I ran into people who came from the information systems from the old days, from COBOL, from databases. And we kind of met in this market funnel, right. Actually, it wasn’t exactly a funnel, the market was expanding. [...] In the beginning of 1990s, slowly, the first fruits of the end of the Market Reserve were started to appear, IT [*informática*] opened, bloomed, companies appeared, the market was growing, right. In 94 it had a different format, a different face. There was now demand for much larger systems, so it wasn’t so much a funnel, it was an explosion.²⁰³

Brazilian IT market went through a major transformation in the early 1990s—a topic discussed in the following chapter. As the restrictions on import of foreign hardware were relaxed, the use of computers grew substantially, creating opportunities for people like Jason. This “explosion” brought Jason in contact with a different world, the one that existed before, but was hidden from him, the world of information systems running on mainframe computers, in many ways representing a different culture. This new world gave Jason some credit for his skills as a “bit twiddler” acquired from foreign books and long ours spent with the computer, at least enough to hire him. The world of mainframe computing (“the information systems from the old days”) was itself in turmoil, facing the opportunity and the challenge of upgrading to the new microcomputer hardware that was suddenly becoming available. It also demanded, however, many skills that Jason did not have.

Jason: I arrived there, migrating, and met those guys who were coming from COBOL, etcetera, right. And it was like: “Oh.” I was at a disadvantage compared to those guys, because I didn’t have theoretical background in data modeling,

203 See appendix C, “Original Interview Quotes” (Jason, June 2007, “The Participants”).

requirement analysis, things that nobody even talked about at the time. So I got in the mood for studying, etc. And went to the University of Estácio de Sá, here downtown.²⁰⁴

This second attempt at university education, however, ended even quicker than the first. Jason was again disappointed by the quality of instruction and was busy with work.

Work inside Petrobras also presented other challenges. As is common in Brazil, Jason was employed in Petrobras as a contractor:

Jason: So, it [Petrobras] had this cycle of public competitions for getting new employees, etc, there are people who are employed by Petrobras. But when it needs something done it gets a person from outside, who works on a temporary contract. However, the law doesn't let you do this for long. Labor laws in Brazil are very heavy. So, what does it do? It contracts an external company, a staffing agency, and tells them: "Hire this guy and sell him to me." So the company hires the guy, he does a contract. But then they have to do a tender, right? So they take bids, another company wins, so "Let this guy go. You—hire this guy over there." So the guy is contracted by three different companies, but really he is just working at Petrobras. They want *me* and they invite the companies to bid on contracting me. [Laughs.] I and thousands of people who worked for Petrobras, to make the process faster, to make things work. Because if they were to depend on opening the competition [for employees], that would take time, a year, two, and then there is corruption, what we call "fish soup" [*peixadas*] right—the politicians picking who will or won't get in, they would game the competition, all of this corruption.²⁰⁵

A year later, one such reshuffling ("Dismiss him here, contract him over there") resulted in a "bureaucratic accident" leaving Jason without pay for two month. While a few months of a gap in payment would shock few people in Brazil, Jason took it as a sign that Petrobras was not an appropriate work place for a software professional: "As an IT professional I am *naturally* averse to bureaucracy. And when it touches me, I get furious. I got ticked off and decided to leave. So I returned to working by myself."²⁰⁶ Jason spent the next ten years working for his own company, finding, like many others, that this gave him more opportunity to practice software development in relative isolation from the Brazilian organizational context.

The contractual arrangement that Jason describes appears to be especially common in Petrobras, since its semi-public status means that hiring and firing employees is even more complicated than it is for private companies, but is often used by many other organizations. Many of my interviewees in fact often told me of multiple levels of contracting. Alta, the software company described in chapter 3.1, provides an example of this. Alta's clients, which include

204 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Fiquei a fim de estudar").

205 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Vende ele pra mim").

206 See appendix C, "Original Interview Quotes" (Jason, June 2007, "Naturalmente averso a burocracia").

private and public organizations, outsource their IT needs to Alta, seeking to avoid the bureaucratic burden associated with having full-time employees. Alta, however, also hires its developers as contractors rather than full-time employees. The developers then form yet smaller companies, typically with just one or two programmers each, which make contracts with Alta.

While contracting appears to be a lot more prevalent in Rio de Janeiro than it is in Silicon Valley, it is hardly unique to Brazil and cannot be easily explained just by the peculiarities of Brazilian labor laws.²⁰⁷ Around the same time as Jason endured his arrangement with Petrobras, “permatemp” workers of Microsoft, one of the most “central” sites of software work at the time, were fighting the company over a similar contractual arrangement.²⁰⁸ Yet, the issue is often understood in Brazil as a uniquely Brazilian problem. The US software working environments are often also idealized in other ways. For instance, they are often seen as places where ideas are judged purely on their technical merit, rather than on personal connections of their originators. Such idealization of the centers illustrates an additional challenge faced by the peripheral participants. When their understanding of how the practice *ought* to work exhibits a clear lack of fit with the local institutional realities, peripheral developers have no easy way of knowing whether the theory carried by the culture of the practice actually fits with the reality elsewhere or simply represents an idealization that should not be taken too seriously.

In part 3 I explore in more detail the many challenges the developers face in making the abstract software culture work in a particular place, looking at three cases that illustrate three of the possible ways of assembling local and global resources for practicing software in a peripheral place. Before looking at those cases, however, we need a better understanding of the larger world of software, its geography, the history that underlies this geography, and in particular how the software development practice has made its way to Brazil. This is the topic of the next chapter.

207 See Barley & Kunda (2006) for a discussion of IT contracting in the US.

208 *Vizcaino v. Microsoft Corp.*, 120 F.3d 1006 (9th Cir. 1997). The case was brought in 1992 by “freelancers” who worked for Microsoft from 1987 to 1990, who seek some of the benefits that Microsoft had offered to its permanent employees, arguing that they had in essence been employees of Microsoft (Harvard Law Review 1997).