TWO

THE POSTRACIAL OFFICE

On a Wednesday morning in June 2002, I was scheduled to meet Jan, a project manager at Globus. I entered the lobby of the glass and steel tower flanking Potsdamer Platz that housed Globus's offices together with two Hyderabadi programmers I knew. The programmers flashed their ID cards while I signed in and showed my passport to the security desk. We took the stairs up to the fourth floor, they hurried to their workstations, and I waited to meet Jan on a low, curved bench upholstered in cranberry-colored velvet. A svelte fortyish secretary in a checkered suit sat behind a blond wood desk, whose organic curves mimicked the shape of the bench. She busily answered the first phone calls of the morning; behind her the company logo—a globe orbited by a shooting star—was painted on the wall in its signature cranberry and gray. As I waited in the lobby, a spot of powder blue caught my eye high up on the wall. In the top left corner above the secretary's desk was a stick-on decal of the head and shoulders of Venkateshwar, the Hindu god easily recognizable by his golden crown and the shield he wears before his eyes to prevent us from being blinded by the power of his gaze. As I pondered how Venkateshwar could have gotten way up there, Jan came forward to greet me. Jan is in his midfifties with a thick shock of black hair, graying at the temples and combed neatly to the side, and a generous paunch spilling over his belted dress pants. I have been introduced to Jan by Adi, who worked on his team before trying to start his own company. Adi had gotten a master of science degree in computer

science at Berlin's Technical University and had coded for Jan for over four years before branching out on his own. He had described Jan to me as a generous and friendly person, and I found Jan easygoing and gregarious. As I stood to greet him, I gestured toward the figure of Venkateshwar on the wall above. He looked briefly up and shrugged, but the secretary, watching closely, nervously bustled out from behind her desk to tell him that she has no idea who put it there, suggesting she was not responsible for it. Jan barely seemed to hear her and just smiled, ushering us toward the hallway to his office.

Jan was not bothered by Venkateshwar hovering in the entranceway yet the deity gave the administrative assistant pause. She was eager to distance herself from this potential matter-out-of-place in front of her superior. Taken together, the administrator's and the manager's reactions encapsulate two different attitudes toward Indians in German offices. The administrator is the face of the company to clients and visitors. She sits in the most designed part of the office, beneath the official logo. Although I did not have a chance to interview her (a sign, I think, of how I was slotted into the office as a researcher meant to interact with other middle-class cognitive workers), the administrator's concern may be read in terms of her position in the office. Though well compensated, her nontech, administrative role and her position at the front desk clearly indicate her comparatively unsophisticated educational background and working-class status.[1] She is unsure of whether she will be made responsible for a potentially unofficial, and perhaps damaging, representation of Globus to clients. Such moments point to class hierarchies in the office. Jan appears to have a more cavalier attitude toward such displays. As I explore below, he thinks of Indians in the office as personally interesting and potentially profitable. Like the cartoon of the plumber and the businessman discussed in chapter 1 (fig. 1.3), this interaction plays out between the two dominant ways that race will figure in the office—as potential threat and as possible resource.

The lobby scene juxtaposes Globus's corporate logo, a globe with a shooting star navigating its equator, with the decal of Venkateshwar, whose gaze is so powerful that if he looked directly at the world, he would scorch it. Globus is a 16,000-person software services company that supports the travel and tourism industry. Globus's tagline, "Solutions That Move You," evokes a world of seamless transitions from one locality to the next. The software that it provides, like the star that travels around the globe, will

link the transnational parts of a business with speed and acumen. Venkateshwar's purview is larger still. Venkateshwar's four hands hold a conch and a golden discus, and he then extends his hands downward in a gesture of protection and surrender. He is the main deity of Tirupati, a vast and wealthy temple in South India estimated to receive 30–40 million visitors per year. How do these two emblems come together within a corporate idea of work and of personhood? What ways of figuring the Indian IT worker emerge and are elaborated on in the space between Venkateshwar above and Globus in the middle atmosphere, above the earth but below the divine? While both Venkateshwar and Globus symbolize success and prosperity, inconsistencies in the way race moves through global spaces emerge when they are brought together in the office lobby.

Analyzing IT economies from outside and as an example of an overall direction in the changing nature of work, theorists of cognitive labor posit a universal, unmarked subject of new economy work—a cognitariat who manipulates signs and symbols through a computer screen. I show in this book that this view fails to capture how such work is embodied. The point is not simply to fill a gap in scholarship by adding in race after the fact but to follow the multiple ways that cognitive work is made real through its embodiment. As Wendy Chun suggests, information "is always embodied, whether in a machine or an animal. To make information appear disembodied requires a lot of work."[2] As the argument of this book continues in this chapter, we will see how racializations are used to create divisions of labor in the office that seem natural—as if emerging from characteristics of the workers themselves. That is, the racialization of IT work is one way that information is made to seem naturally produced without the interference, or messiness, of disciplining bodies. At the same time, and as I argue in later chapters, this materialization of work can free Indian programmers both to pursue middle-class success marked by access to consumer goods, satisfying work, and leisure time and to resist the colonization of life by work.

Race in the Cognitive Economy

In the carpeted hallways of coding economies, discussions of life outside work are as important to managers as technical discussions of process; at the end of the day, cognitive workers often bring work home from the office, and the personality of the worker is parsed by a worker herself and by managers for meaningful and monetizable ideas.[3] Boundaries between

work and leisure are blurred, as a supervisor might come to work in jeans and the office Christmas party takes place at a local bar. At Globus, the rooms are named for prominent German inventors—the Diesel Room, the Zuse Room (after Konrad Zuse, an early computer pioneer), and so on; the transparent walls between conference rooms invite communication. How might the knowledge that circulates around the difference of Indian workers fit into this model of work that fosters communication as a source of immaterial commodities? How might such knowledge also be used to organize workers hierarchically in terms of perceived racial characteristics?

At its core, cognitive labor describes a shift in the locus of production from the production of things to the circulation of ideas and symbols. According to this argument, surplus value is now produced through symbolic systems—such as financial derivatives—that sit atop other financial products. The exchange of these, and other similar abstract (or immaterial) goods, such as Internet page views, online services, and new software platforms, drives global revenue.[4] Following this description of a new economy, it stands to reason that the relationship of the worker to what she produces has also changed. The new economy worker produces ideas, concepts, and abstract products rather than tangible things.[5] For theorists of cognitive labor, the importance of this shift is that even while flexible economies promise an end to alienation, alienation has intensified. That is, since the product of work is no longer made concrete in an object that is separate from the person producing it, the worker is no longer alienated (in the Hegelian and Marxian sense) from the product of work. But, rather than implying a more fulfilling relationship to labor, as one reading of Marx's theory of alienation would predict, cognitive labor entails turning the cognitive capacities of the worker into commodities: "The mobility of the product was made possible by the assembly-line workers who had to remain motionless in space and time. Info-workers, instead, constantly move all along the length, breadth and depth of cyberspace. They move to find signs, to elaborate experience, or simply to follow the paths of their existence. *But at every moment and place they are reachable and can be called back to perform a productive function that will be reinserted into the global cycle of production* [italics mine]."[6]

In other words, cognitive labor at once promises uniting what was once torn asunder—the qualities of the worker and the work itself—and betrays that promise by instead treating the worker's qualities as themselves alienable

commodities. Cognitive labor demands a worker who is always plugged into a network, ready to produce packets of communicated meaning that can be taken up and circulated further in the production of value. In this milieu, knowledge should proliferate by virtue of its links to future earnings potential. Similarly, for management, knowing the internal dispositions of workers is important, not to mold workers into interchangeable factory bodies but to harness their characteristics within a horizon of futurity. As I will argue below, many managers consider the outward habits and appearance of cognitive workers as signs telegraphing their dispositions. As part of what Junaid Rana calls a transnational "migration industry,"[7] race in a cognitive economy becomes a condensed sign of potentials, both acknowledged and disavowed.

Many managers I worked with in Germany expressed surprise and even disappointment that programmers from India did not meet their expectations. Indian workers were unyielding, suggested one in passing; "their mind-set was hierarchical." Another said over lunch that it was necessary for Indian workers to "pay respect to those above them while Westerners were used to working in an open environment where ideas were expressed freely." "I hear," said one Australian programmer in a recorded interview, "that India produces more engineers per year than the rest of the world," and then he went on to be amazed that their quality was so poor. In defense of migrant programmers, another manager of a firm reminded me, "It is only when workers cannot be found for these jobs that we try to find Indian programmers."

These snippets of ongoing conversations about Indian programmers suggest that race functions on multiple registers: as a means of bringing the world outside the office inside its walls as an expression of the personalities of the workers (what we might call "workplace humanism"); as a means of dividing labor in the office according to the perceived abilities of foreign and European or American workers (what we might call a "racialized division of IT labor"); and as a way to allow new impulses into the office that might generate new leads and new creative avenues for producing surpluses (what we might call "racialization as creative content"). Repeated moments of interaction where the strangeness of Indian workers is elicited and commented on begin to make these discourses seem like industry common sense. They devalue Indian coding as "reproductive" rather than productive labor.[8]

In a cognitive economy, race as a sign of human particularity is valued as a repository of human potential even while its salience as a factor in

evaluating work and dividing labor is denied, making the office a postracial work environment. In the following sections I explore the varieties of racialization that cohere around the Indian IT worker. I use the term *race* to indicate when a particular kind of question is raised: namely, is being good at programming (or other skills, such as managing people) fixed in certain kinds of bodies? I follow the different ways that this question is answered, which in their particularities frequently evoke explanations of culture and education. These terms are often ways of talking about how difference adheres in bodies, through training and socialization, in postracial environments.

Postracial discourses construe race as constructed and no longer important—a relic of past eras of slavery, segregation, and legal discrimination.[9] They position the current period as a time after race, at the same time as they allow race to continue to determine conditions of access and inclusion to public goods by valuing "unmarked" success. While most studies of the postrace phenomenon place its emergence squarely after the election of Barack Obama and in the United States,[10] its genealogy in Germany is longer and passes through the history of liberal citizenship after the end of the Third Reich. In Germany today, there are at least three vectors through which German identity passes: the negation of the Nazi past, a rejection of East German history as a history of totalitarianism or the rejection of West German history as a history of capitalist class society, and the use of "the non-German Other to construct notions of the Self."[11] German postracial discourses place racism squarely "in the Nazi anti-Semitic past and in the violent neo-Nazi present, but not [in] racism's institutionalized everyday persistence," as Damani Partridge argues.[12] Postracial thinking in Germany is informed by the use of race in the creation of European identities that attributes both actually existing racisms to nonliberal (often East German) publics and essential otherness to nonwhite immigrant populations.[13]

In IT workplaces, Indian "cultural differences"—a coded language through which to talk of race in postracial offices—is often used as an analytic to evaluate worker skill and as a humanistic sentiment indicating a *weltoffen* liberalism (a liberalism open to the world) as a type of expert knowledge that contributes to workplace creativity. The multiple ways that race signifies in the office are made by the official, postracial suppression of race's continued salience. Culture talk can proliferate freely because it becomes

an indicator of the importance of communication to office work. By writing of offices in Germany as postracial, I intend to broaden this category to include an expansive sense of what race might mean. Rather than describe postracial discourses as a mask that hides the truth of racism, I explore the proliferation of race—the meanings and associations between people, their backgrounds, and the work they do that are mediated by race. These multiple senses of what race means suggests that in the office, the race of Indian coders is used to reflect on knowledge work more generally, as each idea of the Indian IT worker comes with a different set of associations, from "machinelike" to "mystical."

The question "Do work skills inhere in bodies?" is often answered by means of linking racialized bodies to nonhuman things, such as machines.[14] Pointing out interactions between humans and machines has been useful in moving beyond both a human-centered idea of agency and technological determinism. Here, I build on this approach pioneered by Donna Haraway, Bruno Latour, and others to tease out how racializations occur when programmers and programming technologies meet.[15] Following the movement of race across multiple actors, opinions, and demands reveals how racializing Indian IT workers also serves to evaluate the opportunities, difficulties, and demands placed on working subjects in global cognitive economies. Tracing the links between Indian coders and computers is a way to understand human-nonhuman interactions as social processes that draw out potential ways of forming matter and social relationships simultaneously. Various couplings of Indian bodies and computers happen at boundaries and thresholds, where the possible relationship between coding technologies and worker subjectivity is constituted.[16] In the following sections, I first describe how Indian workers create a *weltoffen* workplace. Then, I show how they are racialized in office divisions of labor through comparison with other migrants and with machines. Finally, I turn to how the office mines Indian programmers for the creative content they might provide in the workplace, even while Indian programmers try to discern how to treat these requests for spiritual knowledge in ways that will increase their job security.

Workplace Humanism: The Indian Programmer as Valuable Resource

"I appreciate the cultural things that Indian programmers can introduce me to," said Jan. "They are a valuable resource." We were having a lunch

of falafel, hummus, and stuffed grape leaves at a Lebanese restaurant around the corner from the office. I had spent the morning shadowing him as he sat in meetings and had an all-hands session with his team. Project management at Globus was Jan's second career. He had been a successful real estate agent and made a significant amount of money in speculating on the neighborhoods of East Berlin that became trendy after the fall of the Wall. He used his money to fund a small business services start-up, which closed after a heady year. He was then able to leverage his experience to be hired into Globus. Globus specialized in integrating software platforms for the travel and transportation industries, though its products were also used to manage international conventions and to coordinate complicated international shipping operations. The company had smaller satellite offices in New York, Singapore, and Bangalore. Jan had hired two engineers from India for his team and also sent his colleagues to India to meet with people there. He hoped to increase his outsourcing presence in India. Jan felt that he knew quite a bit about India from employees. He pointed out that Indian engineers can talk to the people in India in their own language and make everything work faster. They also provided valuable information on the Indian mind-set and the Indian market, which he glossed as the *etwas anderes* (something different) that he really enjoyed talking to them about. Jan was glad he hired programmers from India, not just because of their coding skills but also because they brought exposure to a different way of life.

According to programmers who coded on his team, Jan was fond of popping into their work area, asking such things as, "Why do Indians worship monkeys?" "What is the significance of 'red' in Indian culture?" and "What does the red dot on the forehead of Indian women mean?" The queries were confusing to these programmers because they could not puzzle out the relationship between these questions and his job, which is to evaluate and guide their work. Jan thought of his interest in India as enhancing workplace culture, as an expression of his desire to be open with his workforce and take an interest in their lives outside of work.

Andrea Muehlebach writes of morality as part of the structural transformations of neoliberalism, inserting selflessness into the heart of capitalism.[17] The "something different" that Jan said made Globus a cutting-edge place to work also comprises a moral project making him a tolerant and world-open German citizen and creating a global office where differences

are happily accepted. Jan's questions demonstrated that he was a liberal German subject. Such liberal tolerance of non-German others is one way that German citizens can show distance from the race hatred of the Nazi past, the East German totalitarian past, and the West German class society.[18]

Outside the office, programmers discussed how to respond to Jan's questions about India. It turns out that many other programmers have heard similar questions, if not from managers, then from curious coworkers. One programmer told me that "it is embarrassing that they seem to know so much about India," especially when these are things that Indians themselves do not know as well. Another said that it is "baffling as the questions come out of nowhere and it is hard to respond," so he just agrees with whatever the person who is asking thinks. A third recommended trying to give a simple answer, with which the questioner is mostly satisfied.

One thing that most managers and programmers explicitly agree on is that background (nationality, gender, race, and so on) should not matter. All that should matter is that a person has the skills to get the job done. Yet, talk about religion, symbolic meanings, attitudes, and predilections swirls all around the workplace, and especially around Indian programmers. It is unclear to most programmers why this continues to be so. Some see the curiosity about India as evidence of world openness, as Jan does. Some see the questions as evidence of the superior knowledge of their European colleagues, who seem to know so much already about India. Some put it down to simple curiosity, while others think that the questions are a test to see how well they can adapt to a Western work environment, coded as transparent and nonhierarchical. These instances of cultural curiosity and the intense strategy sessions among programmers that they spark speak also to the way a picture of Indian difference builds up over time, through a series of individual interactions, and then condenses into a sign of the likely proclivities of the Indian workforce.

Racialization as a Division of Labor: Indian Coders as Uncreative

Michael was in his midthirties and had moved to Berlin five years earlier when I met him in 2003. He had close-cropped hair and square, black-framed glasses. He wore high-end leather tennis shoes and expensive jeans with button-down shirts. He loved obscure 1970s psychedelia, and when he was not in the office, he would take me to record stores in Kreuzberg, on Savignyplatz, and in Prenzlauer Berg looking for rare pressings. Michael worked

for a small start-up that made translation software for government publications. The company was hoping to pitch its product to the EU administrative units in Brussels. While combing the record bins in a tiny, dusty shop in Pestalozzistrasse, Michael broke down for me the organizational divisions in his small office in the trendy neighborhood of Prenzlauer Berg. First, he outlined the difference between those who answer how-to questions about software packages over the phone (tech support) and those who write the source code of a project (software developers). Tech support commands neither the same salary nor the same knowledge as developers. He characterized this as a general divide between degrees of skilled labor. Another type of divide—between soft and hard skills—complicated this picture. Pulling out a copy of an early Pink Floyd album called *A Saucerful of Secrets* with a boyish grin, he continued, "Most offices are divided between front-facing areas for interfacing with clients and back-facing areas where problems are solved. These are called front-end and back-end functions." Front-office people, he went on to tell me, were said to have good communications skills that back-office people lacked and therefore commanded higher salaries and comparatively greater job security. Finally, there was the issue of an employee's potential—what the programmer may bring to the company in terms of possible future earnings. In Michael's firm, the Indian programmers hired to do software testing were often included in meetings where, in Michael's opinion, their expertise was not really applicable. Michael suggested that his boss was hoping they might chime in with what he called "cultural knowledge" (*kulturelles Wissen*) that might help him the next time he needed to hire a short-term Indian programmer. Michael did not fear that one day his job would be done by a migrant programmer or that it would be outsourced, for he knew his place in the division of labor outlined above. He was a "creative," a project designer who did upper-level, front-facing work, for which he was uniquely skilled by virtue of his schooling, which stressed creative problem solving, and his "interpersonal communications" skills (his term), which ensured he would interface with clients.

The inclusion of Indian programmers in meetings, which could be read as a sign that they were being groomed for management, had another function according to Michael. They were there as reservoirs of cultural knowledge that might be tapped in future management operations. I was surprised at how often Indian programmers were included in

management meetings and often asked about it. Sometimes, this was part of training for management, but often, firm managers had other purposes in mind.

In an interview with one project manager at the annual technology conference in Hannover, Germany, in 2006, I asked about the inclusion of his Indian developers in these team meetings, called "scrums" in many firms. He answered by revealing that his firm was hoping to build another satellite office in India, and to do so, he needed to train Indians "to be Indian but to dream in German." He unwittingly (most likely) rehearsed thereby an earlier moment of elite Indian class formation, when British colonial policy inclined toward producing a cadre of Indian civil servants who were "a subject of difference that is almost the same, but not quite."[19] As Thomas Babington Macaulay, writer of *Minute on Indian Education*, could not help but imagine in 1835, "A class of interpreters between us and the millions whom we govern—a class of persons Indian in blood and color, but English in tastes, in opinions, in morals and in intellect."[20] Though, in this case, the taste, opinions, morals, and intellect would be German.

For some, culture talk makes a workplace global. For others, culture talk—when carefully parsed as talk about education or family values, not biological essentialism—is about learning to better manage a culturally diverse workforce. The patterning of ideas about Indian programmers cohered around the limited ability of Indian programmers to intuitively grasp Euro-American standards of bodily comportment and office etiquette. It justified their relegation to short-term jobs and repetitive coding. Importantly, these opinions were shared by Indian cognitariats who had reached the upper echelons of management. The following conversation is representative of conversations I have had about Indian knowledge workers with regularity over the past five years, in Seattle, Mumbai, and Berlin and even in airplanes hurtling through the skies somewhere above and between these places.

"The IT industry is ruining India," exclaimed Rahul, the ice in his whiskey and soda clinking thickly as he gesticulated. "The salary for IT work is so high," he told me, "most comp sci [computer science] students do not go on past their undergraduate degrees. India is not emerging as an R&D center the way it could be." I was on a plane bound for New York, sitting next to this Indian Institute of Technology (IIT)–trained computer scientist who lived in the United States and worked in the computer science department

at a large state university. I took notes on our conversation as he talked, telling him I was writing a book on upward mobility in the IT industry and soliciting his opinion. Rahul traveled frequently between the United States, Germany, and India on research and recruiting tips. He was frustrated with the majority of Indian technology workers. While he understood the allure of a good paycheck and told the undergraduates he met in India they were right to pursue these jobs, he felt stymied by the lack of innovation on the part of Indian engineers. In the end, he complained, echoing other managers I meet, the Indian computer programmer is diligent but not creative. "I use the analogy of the architect and the stonemason," he said. "The Indian engineers are very competent stonemasons but do not become architects." Rahul blamed the education system. A government stipend for graduate education is paltry compared with what a corporate job would yield. I asked him why Indian companies do not try to fund education, he thought a moment and answered, "I don't know. They are making so much money doing what they are doing, they don't see the need to make architects."

In a famous passage in *Capital*, Marx compares architects to bees: "What distinguishes the worst architect from the best of bees is that the architect builds the cell in his mind before he constructs it in wax."[21] Rahul, when comparing architects and stonemasons, echoes this sentiment, calling out the Indian education system for being content to train engineers not to think but simply to reproduce pre-given forms. Marx points to a distinction between plan and instinct. When man labors through plans, he "develops his slumbering powers and compels them to act in obedience to his sway," writes Marx, in an exposition of how humans not only affect their environment but also can purposely affect human life, or species-being, itself.[22] Rahul instead reasserts a division within cognitive labor between those who are assimilated to planners and those manual laborers who are merely executors. He folds Marx's call for denaturalizing the conditions of life into a Smithian naturalized division of labor.[23]

A French Canadian director of sales for a French company in a thirty-person office in Berlin, with primarily German and Eastern European clients, whom I meet at a regional trade fair, had an even lower opinion of the Indian engineer. His company makes a piece of software that, as he described it, "sits on top of another software application," making it customizable for the needs of various companies. He was regularly asked to support their product when companies cannot resolve issues themselves.

In this capacity, he often dealt with outsourcing operations in India that provide technical support to these companies. The technical support teams simply forwarded queries about the software to him, without first finding out if they could solve the problem on their own. "Indian IT people need a lot of hand-holding," was how he puts it to me when I interviewed him one day in the offices of his firm. In his opinion Indian programmers were not very good. In his experience they did not try to solve anything themselves; they simply passed it on to someone else and waited for the answer. I asked him what he thought the outcome of the German green card program would be, given his low opinion of Indian programmers. Perhaps bringing them over here for training and then sending them back to India, he answered, would be better because they would learn how to think and do on their own. He used an analogy, telling me that software development and maintenance were like designing a car and being a car mechanic. The designer was the one who makes the new Ford, whereas the mechanic keeps it running and repairs it. "Most Indian developers are more like car mechanics," he concluded, "than like designers." Indian programmers figure as problem workers who lack creativity and initiative—as lower-class workers who are car mechanics, not designers.

Analogies like the architect and the stonemason and the designer and the car mechanic downgrade Indian expertise. In a meditation on how the question of intelligent labor played out in relationship to technology, Simon Schaffer notes that the question of human and other kinds of intelligences was long used as a means of marking nonwhite, non-European populations: "In many western myths of mechanical intelligence, with Chinese or Japanese, Turks or Nazis as their protagonists, aliens are automata, mindless subjects of tyranny; they build automata, because they possess fiendish cunning. . . . There is, perhaps, a long-term political and aesthetic relationship between intelligent automata, orientalism and the covert."[24]

Mechanics, stonemasons, and bees all possess a craft but not a plan. They may be cunning but are not creative. As metaphors for Indian workers, they provide an example of the way unofficial discourses of race circulate in a postracial office, where racialization naturalizes division of cognitive labor, embodying certain forms of work in particular, nationally branded kinds of workers. These figurations of the IT worker help create them as essentially different, because their intelligence is outside the ambit of the human intelligence of the architect or car designer and represents instead

the instinctual, otherworldly intelligence of the bee or the untrustworthy intelligence of the car mechanic. Indian programmers who have a mechanical intelligence may be incorporated into the corporate office, but forever in a subordinate position.

Exposure to an Indian workforce is highly prized by many managers. Being able to demonstrate effectively that they are culturally confident makes them globally competent. Because of the potential that has been invested in Indian IT as a source of new consumer markets and cheap labor, some IT managers are proud of contacts with and abilities to deftly navigate the "difference" of India. Jan, the manager in Berlin with whom this chapter opened, boasted of being sufficiently versed in Indian culture to feel that he could at least tell to whom he was talking when he worked with people from India. "I can hear the difference between Hindi and Urdu," he said, telling me that he can differentiate between the two South Asian languages often associated with Hindus and Muslims respectively.[25] He further indicated that he cannot "distinguish between all the dialects, like Tamil." It seems almost impossible that Jan could pick up the difference between Hindi and Urdu but not between Hindi and Tamil (which is a South Indian language that sounds quite different from North Indian languages). With his evaluation, he brought home the importance of being able to claim cultural knowledge for the sake of being a good manager. At the same time, though, he also made clear that for him as manager, a deep understanding of differences within India was unimportant. That he could leave to his Indian subordinates.

Often, when discussing the particularity of Indian (and sometimes also Chinese) programmers, managers and European and American programmers would stress the university system. At a cocktail party in the United States after I had returned from fieldwork, an American programmer responded to a description of my project by saying that he notices Indian and Chinese programmers are mathematically minded and think of code as math, whereas American programmers think of it as a language and can therefore be more responsive to client demands. He suggested that checking in with the client more often—what is called "agile" programming—can avoid delays and miscommunications. But, because most Indian and Chinese programmers come out of a regimented university program, they are not taught to think that way along the lines of communication. On the other hand, according to this programmer, many American and to some extent European programmers are self-taught and therefore have a

different approach, treating code as language, which to him meant something that is naturally communicative, there to be played with and improvised on.

Yet another way to divide the architect and car designer from the bee and the mechanic, the division of coding into language and math makes concrete a creative and repetitive approach to programming. The "East," defined by rote learning, is limited, while the "West," defined by poetry, expands. These differences, in keeping with a postracial imaginary, are carefully couched in national-cultural traditions of training, which allows them to be all the more effectively generalized. Such metaphors of difference, employed to explain to an ethnographer what the differences are between foreign and native coders, provide a frame through which participants in these worlds can explain and naturalize differences in experiences in the office to themselves and to others. As partially expressed codes for divisions of labor, they at once justify hierarchy and maintain the illusion of workplace meritocracy—where the best skills are rewarded by the best paychecks. They do so by expanding the scope of skilled labor to include the "soft" skills of human interaction and the "hard" skills of creative coding, thus rekeying the work of programming in two modes from which foreign coders are normally excluded.[26]

The Tale of the Turkish *i*

Within the framework of liberal tolerance, only a limited spectrum of behavior can be accepted without fundamentally threatening the stability of a democratic nation-state built on the idea of freely choosing and rational individuals.[27] In the case of Germany, Turkish immigrants are repeatedly framed as violating the principles of democratic society through their religious beliefs, "refusal" to speak proper German, putative associations with Islamic terrorism, and patriarchal practices toward women.[28] Indian migrants, on the other hand, may be framed as acceptably different.

A project manager named Björn, whom I met in the Mitte offices of Dash Technologies, a firm that builds specialized software to work with existing business platforms, worked on internationalization and localization, also sometimes called *globalization*. His team was charged with making all the programs produced by a parent company work across multiple languages, ensuring that there will be no mistakes or complications when keyboards, interfaces, and design elements move from one context to an-

other. Because of the international nature of the work, he has built a team from South America, Russia, India, and China. When I asked for clarification by example of this programming of global fluidity, he began with an old case, he said, that still works as a good illustration:

> When an e-mail program was first introduced, it used the icon for a mailbox that you would click on. The icon was round on the top and flat on the bottom, rendered with some degree of perspective. But this icon, which came from an American company, did not work well at all elsewhere. They rolled it out in Germany, for example, and no one would click on it. They all thought it was a breadbox. Eventually, after trying many other things, a group came up with an icon in the shape of an envelope. That was universal, worked everywhere, and people clicked on it.[29]

Localization and internationalization are about making programs work properly across different cultural and linguistic contexts, as the mailbox example suggests. For Björn, it was also a perfect location from which to notice differences in capacity parsed by nationality. Björn provided a second example but started off from quite an unexpected direction. "Indians," he exclaimed, "are so good at programming, but they will never ever have an internationalization center in India." In his opinion, Indians just are not interested in translating programs into other languages, not even their own regional ones. "Maybe it is because they are all so good at English, they think everyone should just learn that." Riffing on these problematic cultural differences, he moved on to the story of the Turkish *i*.

"This is an infamous problem because of what happened to a well-known cell phone provider. In Turkish, there are two different *i*'s, one with a dot on top and one without. A certain phone company did not make note of this difference when translating the texting software from English to Turkish." This company, which Björn did not want to name, was not even aware of the potential problem. They translated the text recognition software into Turkish so that the phone user could use predictive text software, which suggests words to the user as the user types. They only put in one *i*, even though users would clearly need two different kinds. "The issue is really quite serious," he concluded, "since if they used the wrong *i*, the sentence's meaning would completely change." In the project manager's story, this is exactly what happened. "One day, a man received a text that was supposed

to say one thing but said another." I forget what the mistake was, Björn told me, "but it was bad because the man who got the text went into a rage and almost killed someone."[30]

The tale of the Turkish letter contains within it many kinds of stories; important among them are two cautionary tales of failure: one of the Indian coder to consider internationalization problems, and one of the Turkish cell phone user to act rationally in the face of an internationalization error. These two figures—the Turkish migrant worker and the Indian temporary coder—play off against each other as counterexamples of the productivity and danger of migrant cultures.

As if to underscore the rightness of his reading of Indians as good at code but not good at cultural translation, Björn told me about his old job, when he worked side by side with a developer from Tamil Nadu (in South India). It was 2004 when the Indian Ocean tsunami hit. He watched the coverage on the news and knew that his colleague was from an affected area. He went into work the following day and asked if everyone in his family was all right. His colleague seemed surprised. Yes, he answered, everyone in the family was fine, why was he asking? Björn said, because of the tsunami of course. His colleague told him that only the poor people suffered from the tsunami, and he should not worry. The poor people, he said, were not really full people anyway. Björn was shocked and dismayed. This was a sure sign that Indians did not really care too much about such things as the digital divide. Whatever might be said about the smug indifference of the Indian programmer toward the victims of the tsunami, for Björn, this was one more piece of evidence of insular thinking that helped him decide on the worth and the correct placement of his highly valued Indian team members.

Indians as Automata

On an Internet forum for German-speaking businesspeople, participants discuss the pros and cons of working with Indian programmers. Under the heading "Indian IT," a project manager relates a story of work that was outsourced to Mumbai. I excerpt a lengthy passage from this post to give the reader the narrative arc of postracial justifications of essentialization:

> I was asked to test a program that was allegedly already developed. The program was supposed to order financial information into a table, and

then as new material was entered in the table, automatically refresh each field or leave it as it was, depending on the calculation. My first test showed that the refreshing of the table worked correctly, but the data that was already entered into the table vanished when the table was refreshed. So, I did not accept the development and sent the whole thing with detailed reasons (in English) back to the developer in Mumbai (India). After a few days I received his extensive analysis of the problem with the conclusion that the functionality I required was not described in the original development contract. Therefore I should write a new contract with the required development described in it. "My good programmer" also cc'd my manager on this email. Everything must be done correctly! [*Auf cc hat mein guter Programmierer auch meinen zuständigen Manager gesetzt. Es muss ja alles seine Richtigkeit haben!*] In my answer I explained to him that he disturbed through his programming a functionality that was already working. He should simply program in such a way that what was already there remained undisturbed. . . .

I took some time to try to figure out if we were simply not understanding one another [*aneinander vorbeireden*]. But since I have been working on international teams with English as the project-language for at least 6 years, I had confidence that I could explain difficult cases in English. After a few days I received an email in which he questioned what he already had programmed. The point was that the programming contract did not describe what he was supposed to do with the data that was already in the table. He had programmed correctly to begin with, but now that the situation had changed, he had to start from the beginning again. This whole process had already taken 3 weeks. I then asked him as cynically as it was possible for me to in English, why he thought that it said that his program should calculate something. And why it did not say that the program after calculation should not do anything else. And if he had ever heard that sometimes circumstances are implicit in the description, without them having to explicitly be formulated. He agreed with me. After two more days of detailed explanation through Chat, he began working again. This part will surely be in the bill including these two days. I spent 3 or 4 hours on this during which I had other things to do. . . .

The development time [*Entwicklungszeit*] has now been extended by 8 weeks. How many (cheap) hours will be charged by the Indian company I do not know. For my part, I needed about double the amount of

> hours that I would need if I had been working with a halfway accomplished programmer in Germany. . . . Granted, this is an extreme example. For me, it is not necessarily a question of communications difficulties or cultural differences [*Verständigungsschwierigkeiten oder kulturellen Unterschieden*]. In the years in which I have worked with offshore programmers, most of them were just out of University and could program ABAP [Advanced Business Application Programming, used by the German firm SAP] but had no idea about business processes. . . . There are also good Indian programmers. But they usually work the same hours as we do, not offshore anymore, but here with us in Europe or the USA.[31]

The author of the post simultaneously masses his impressions around the poor working habits of specifically Indian IT experts while denying that culture or his own biases plays a role in the interaction. He relates the story as one of initial open-mindedness that was slowly replaced by frustration—until he has to intervene by using sarcasm to make the programmer work as expected. In the end, to forestall perhaps the charge of prejudice, he claims that there are also good programmers from India and separates those who are in India from those who "make it" to the United States or Europe.

The behavior he describes is echoed by those who respond to the post. Four individuals wrote lengthy responses to this post. The first response makes it even more clear how an image of an Indian IT worker who lacks sophistication and is more machine than human develops: "Indians—sorry if I generalize—only do exactly what one tells them. That is, they execute only like machines, no more, no less [*Sprich sie führen wie Maschinen nur aus—nicht mehr aber auch nicht weniger*]!!" Again, the writer is careful to mark his awareness that this might come off as a generalization but goes ahead and asserts the machinelike quality of Indian programming. This form of discourse at once rehearses the German overcoming of prejudice and elevates the author to "the ideal translator/interpreter of contemporary and past racism."[32] In this way, the judgment of Indian IT workers escapes the accusation of racism even as it uses probabilistic qualities of personhood to make general pronouncements.

Another response suggested that the problem was that the programming requirements were not written precisely enough. The writer gave the example of how the distinction between male and female could be ren-

dered in several different ways: m/f, m/w (for the German *männlich* for male and *weiblich* for female), 1/0, or 1/2 and so on. Not specifying in the requirements exactly how this was to be written resulted in huge problems later on. This analysis is important because it moves in a different direction from the main thrust of this post, suggesting that responsibility lay with German assumptions about language and management practices that shifted blame to the overseas, outsourced workforce. Unfortunately, it remains an anomaly in a field otherwise crowded with repeated statements of the difference between those "over there" and us "over here." The final two responses add economic reasoning to the East-West dimension of these divides, noting that for big European and U.S. firms, using Indian programmers was a way to sink costs, while Indian firms soon realized that they could train their own people on the Western firm's dollar, and so they continued to send over their new people when they had to do on-site work.

In chapter 3, I show how Indian programmers working in German offices try to make sense of the demands of their managers at the same time as these workers criticize their treatment in the office. The perspective of the programmer in Mumbai who is at the other end of this story is not available to us. Yet, the second response I discussed contains a possible alternative transcript of events, in which the protocols are written unclearly, so the programmer tries to do only exactly what is written and no more.

The All-Hands: Racialization as Creative Content and the Dream of Replaceability

Back in the IT offices of Dash Technologies, we were called for the weekly all-hands meeting to the Oahu Room. The room names, each one for a different Hawaiian Island, fit into the overall ambiance of the workplace as a space that fosters imagination and creativity. The team members entered the room and joked about leaving cold Berlin for the islands—that they should begin to decorate the rooms with tropical drinks and palm trees. They are all looking forward to what Björn had christened "Pfannkuchen Freitag" (Flapjack Friday), when the team would roll up their shirt sleeves, remove their jackets, and spend an hour in the morning making pancakes for the office. This Wednesday, the all-hands meeting concerned the rollout of a new software product that had to be launched simultaneously in multiple European countries. As the project manager reviewed the progress his team has made so far, he repeated the team slogan for the project, "The

Internet is connecting computers, language is connecting people." Björn looked around the room and addressed each member of his team. He had staff from Russia, the United States, India, Spain, and Portugal in the room that day. In each case, he told them, they needed to be responsible for making sure there were no mistakes in the translation of the software into their respective languages. For my benefit, he asked two of his staff members to outline past mistakes in other projects. In Argentina, for example, the global version of a world map software application was translated into Spanish. But the Falkland Islands were marked as British in that version. Björn suggested that the head of the team in Argentina went to jail for treason over this. Another case occurred because someone "forgot one pixel" and the map of India left out Kashmir, but when the map of Pakistan came up, Kashmir appeared as part of Pakistan.

Asking a Hyderabadi programmer on Björn's team what she thought of her colleagues after the meeting was over, she said she "loves the work" and considered "the problems with global rollout fascinating." She was, echoing all the other team members with whom I spoke, enthusiastic about the team's international makeup. Being able to work with people from so many different countries was one of the primary perks of being in global software. This programmer had greater goals than being only a subsidiary member of the team. For her, too, international exposure would provide another reservoir of expertise in cultural differences and the international protocols of corporate programming work. As many Indian software developers told me, international "exposure" made them more attractive hires back in India. She eventually wanted to be where the project manager was sitting. To that end, she had bought a book to teach herself German at night. She recognized that although the team really needs her right now to translate between German and Indian offices on this part of the project, once the Indian office understands the workflow, she would be obsolete. She was eager to prove herself useful to the team for now by providing information about India and hoped, as they kept turning to her for this knowledge, she would be able to grow and expand her knowledge of management practices in a global setting.[33]

This striving may not yield upward mobility. Perhaps the greatest sign that she and other Indian coders will have a hard time moving up from their temporary positions came from Michael on one of our record-finding jaunts. He told me that the "holy grail" in the industry right now is to use

social networking and crowdsourcing to do the work now done by Indian programmers. He himself was skeptical that this plan would ever come to fruition the way companies think it will, calling it "turning iron into gold." But the idea is that costs can be cut even further by using machine translation together with community members who would be offered discounted rates on the software for their services. The machine would translate, or produce batches of code, and then the translation, the original, and the source code would appear on a screen side by side. A community member would compare the two translations and correct the native language one, and then other community members would review the source code. When the same correction was made at least four or five times, the machine would learn the correct phrase and incorporate it.

Eventually, the Indian programmer as well as foreign language experts would be competing (and losing out against) such cost-effective models. In this future vision of labor, migrants are replaced not by a solitary machine but by a network of human machines. So far, the Indian coder, especially the one working in India, is still less expensive than the machine-human network.

In knowledge economies, the world outside the office is brought inside, precisely because the new economy relies so heavily on deep reservoirs of creativity: "Labor-power increases the value of capital only because . . . [of] its inherent connection to a productive cooperation richer than the one implicit in the labor process.[34] Because "culture" is a resource in the office, Indian programmers have a privileged place at the conference table. They are relied on as a crucial intermediary between Germany and India and are a valuable resource in producing new communicative competencies that might one day open India as a new market for the company's products.

At the same time though, the repeated layering of the inadequacy of Indian programmers for the sophisticated "culture work" that is required in translation projects creates an unofficial division of labor that keeps most Indian developers in the back office rather than moving them up as project managers who would command a higher salary and a permanent position with benefits.[35] The explanation of communication as fallible proffered in one of the discussion board comments is rarely taken up. Instead, most office communiqués operate as if there were a one-to-one correspondence between words and their meaning. Despite an awareness of the difficulty of translation as software moves across language zones, in interoffice

communications, the reigning language ideology tends toward sense-reference predication, because everyone is speaking the same language, international business English.[36]

Modern types of office work appear interchangeable from the outside, because most cognitive workers sit in front of computer screens and keyboards. But in actuality, skills are more specialized, less easily learned, and less interchangeable than ever: knowledge workers "could never exchange jobs since each and every one of them develops a specific and local ability which cannot be transmitted to those who do not share the curricular preparation and are not familiar with the same complex cognitive logic."[37] The increased specialization is, in part, an effect of folding workers' outside interests into work itself so that workers are invested in work as never before.[38] The boundary between the work self and the home self blurs as workers' personal resources are drawn into the circuits of production. But it is precisely in drawing the biopolitical capacities of human life into the work process that workers become simultaneously individualized and *dividualized*, in Deleuze's terms, both more like themselves and singular and more a representative of a population to which they are made to belong.[39] Thus, while Indian programmers become less interchangeable and individually more specialized in their work, they also become a *type* of worker who fits in a shifting slot in IT economies.

Inhabiting Human-Nonhuman Interfaces: How to Be Safe on Unsure Terrain

One evening, as I sat with him in the darkened offices, Mihir, a twenty-five-year-old coder and business graduate, turned to me and said, "The work is quite boring, you know—*just waiting* for some client to call in with a problem." Looking around the empty office, he began to link his work to his personal timeline. He grew up in Bombay (now Mumbai) as the son of a government worker and a pharmacist's assistant. Trained at the University of Bombay in computer science, he counted himself among the first recipients of the German green card. First, Mihir took a job with a small company in Regensburg, a city south of Berlin in the state of Bavaria, and then moved to another small company on the outskirts of Berlin after his six-month work contract expired. According to Mihir, Berlin is a big, open city, teaming with life, though not too full, like Mumbai. But Mihir realized that the imminent end to his current project meant seeking out new opportunities yet again.

Mihir's job was to test applications for a cell phone company. The company's ringtones had not been uploading correctly, leading to many customer complaints; he was on call to respond to them and fix the problem. Long after everyone else has gone home, he sat in the office hunched over his keyboard comparing lines of code across open windows trying to find the problem. He leaned back wearily rubbing his eyes and correlated the repetitive fixes he makes with the steps in his own timeline:

> I was always good in maths. I went to King George School near my home and sat on the same bench as the guy who is now an important man at Wipro. At that time, I was not focused. I used to go and feed the stray dogs after school instead of studying. They were dangerous to others, but they never bit me until much later. Once when I came back home for vacation, I got bit. I got top marks and was thinking of studying architecture. My father was not against the idea but thought I should first study engineering for practical reasons.

Mihir did study engineering but was tired of school after so many years of studying hard and after-school tuitions (tutoring), of keeping up with his friends, and keeping his eye on his practical skills. He thought that when he finished this degree he would take some time to do something else and maybe return to architecture. But when he was finished, he noticed that all his friends were going for higher degrees because they would get better jobs and higher paychecks. "So," he said, "I went in for a business degree." While he was completing his MBA, his parents and some of his friends suggested he specialize in computers, since he was always so good at math, and in business school he specialized in technology and computer management.

On the weekends, Mihir has a ready smile and is brimming with talk of the latest Bollywood release, his next visit home, a new recipe his wants to try, or a restaurant he would like to visit. But at work, his spirits visibly sagged as the night wears on, his shoulders drooping and his skin assuming a pallid, jaundiced cast.[40] I asked Mihir, since he has not had formal training in computer science, how he knew so much about coding. He said he had picked it up along the way, because he had always been hired in development jobs rather than in management jobs, despite his management training. He compared himself with his friend from school, who got the job at Wipro. He had a straight course from school through computer science to

Wipro. But because Mihir was not focused and pursued (according to his own reckoning) meandering courses through life, he had become stuck as a low-level engineer.[41] He sounded tired and frustrated—just the opposite of how he is on weekend outings when he extols the pleasures of open spaces and easy sociality as an antidote to the stresses of the workplace. Yet, he also seemed to suggest that a return to his carefree boyhood days was no longer possible. The strays who never used to bite him now do, perhaps sensing he no longer is native to their haunts. Mihir has continued to adapt to current industry needs, remaking himself through his degrees into the perfectible white-collar brown-bodied laborer for the global economy.

One evening, he told me about the atmosphere during the day when his German coworkers and boss are around. He has made good friends with Sasha, who shares his worktable, sitting across from him in the hushed atmosphere of quiet concentration that pervades the floor. They go to movies together and have dinner; Mihir has introduced him to Bollywood film and told me that his friend bears a strong resemblance to Saif Ali Khan, a well-known Bollywood star. But his boss, said Mihir, was a different story.

Although at first the boss had been friendly, open, and curious about the lives of Indian programmers, lately he had begun to lecture them on etiquette in the shared kitchen. Mihir was hired as part of a two-person team from India; the other Indian programmer was an excellent coder but did not speak English very well. Mihir was often asked to explain things to him, including one day when the boss decided to fine this programmer for breaking a dish in the kitchen. Mihir thought that this action was over the top, but the boss started correcting their behavior repeatedly, telling them how they must correctly load and run the shared dishwasher. "I don't know how they do things in India," said Mihir's boss, "but here in Germany we place great emphasis on keeping things clean." For Mihir, these missives were not lessons in how to be a good coworker in a global workplace. Mihir understood these as signs of the boss's true nature—nice on the surface, but ruthless underneath. In the same context, Mihir told me that his salary was actually less than he was expecting because the firm was located just outside Berlin in the town of Potsdam, where according to labor laws, salaries could be set lower because the city was in the former East Germany.

Managing in the office requires "an *expertise* of the personal dimension of work. To administer work, it has become necessary to know, to calculate,

to deliberate, and to evaluate."[42] I would add that knowing and calculating capacity based on the signs of race is part of this work of expertise. Race as an embodied marker of cultural difference is both valued and confining in the IT office. For the Indian programmer in Björn's office, this leads to a permanent remaking of skills in an attempt to add value to the office because of her race and to transcend the end of this knowledge's usefulness. In Mihir's case, the racialization of cognitive work leads to a profound destabilization because his boss appears to switch suddenly from opening and welcoming to punitive. As I have argued throughout this chapter, in the German context, talk about ethnicity is both outright and hidden, it is at the same time about the different qualities of different workers and about a general sense that cultural differences in the workplace reconfirms German tolerance and opens up new markets to business expansion.

A project manager who travels between the United States and Germany and is of Indian descent viewed these practices from the perspective of North American histories of discrimination. She told me that when she visits the offices of her multinational corporation in both countries, she is appalled by the way the Indian IT workers are treated. They are quite literally, she told me, put in a back room, hidden away from everyone else. She always tries to treat them as human beings, though, just in the way that she talks to them. After thinking for a moment, she admitted that they have no idea how they are coming across. For instance, they think nothing of bringing food from home and heating it up in the microwave, regardless of how it must smell to the others. She argued that to move out of the back room, Indian programmers would have to give up their habits and become aware of how they must appear to others. They would have to learn their footing on a terrain of useful and unwanted differences, what Smitha Radhakrishnan calls "cultural streamlining," if they are to move up before being moved out.[43]

In an online article posted on a website for Indians living in Germany, columnist Usha Amrit counsels flexibility as a response to unfamiliar workplace demands:

> In company recruitments in Europe a lot of emphasis is given to a potential candidate's social persona rather than just his or her mere qualifications and professional experience. I suppose, this takes more precedence if the candidate is an immigrant or foreigner. We [Indians] possess a certain inherent flexibility when it comes to working in diverse

environments. After all, it is not uncommon to find oneself working with colleagues from various parts of India, on an average in any work environment in India. For instance, your Tamilian boss pairs you up with a Bengali colleague and under your tutelage is a Bihari who works in conjunction with an Andraite while you are working on a project outsourced from a European company. . . . Working in an entirely foreign environment also requires knowledge and training of working styles that vary from country to country. However, I'd say given a rough survey of people across the world, we Indians with our innate flexibility are most likely to be at ease in having ourselves transposed and transported (given similar salaries and perks of course) to totally foreign work environments.[44]

The conversations and advice Indian programmers give each other on how to respond to questions about culture, questions of etiquette, and awareness of smells that might give offense are examples of cultivating what is taken above as an "innate" flexibility. They evaluate the "essential" quality of being Indian positively, all the while remaining within an essentializing discourse. Amrit's advice to roll with workplace norms so long as salaries and perks are adequate reminds me of Mihir's problems with his boss. He should obey the demands on kitchen etiquette but also look for another job in Berlin where his salary would increase. This type of advice, of course, ignores the way race is speculated on to produce both surplus value and a profile of employee worth more broadly across corporate software industries.

Producing Difference as Supplement: Rethinking Race in a Global Economy

Potsdamer Platz, where once stood a vast no-man's-land between East and West and earlier still Weimar's Mecca of dance, entertainment, and nightlife, now is home to looming corporate headquarters, a shopping mall, and a few remaining fragments of the Berlin Wall that gesture faintly at the immense divide between the socialist and capitalist Germanies that once defined this city. On one hazy May morning, I met Sasha at the exit of the U-Bahn on the west side of the large and busy square. Sasha was Mihir's friend and colleague from the office who really did look like a German Saif Ali Khan. We were not meeting to talk Bollywood movies, however.

Nor were we going to rediscover the art nouveau splendor of late nineteenth-century buildings like Weinhaus Huth, almost destroyed by Allied bombings. We were instead on our way to a conference on outsourcing, migration to Germany, and Indian coding expertise sponsored by Daimler-Chrysler and held at its gleaming corporate headquarters on the rebuilt city square. All the big-name German IT firms, including SAP and Siemens, would have representatives there. I was curious to see how they would present the green card and Indian IT to their constituency, an assembly of business professionals and local reporters. Sasha had been sent by his company to this IT fair to represent the company and network with other German IT professionals.

As we walked through the cool, damp air of a spring morning, I asked Sasha to tell me more about his background. He grew up in the West German town of Kaiserslautern (home to a U.S. Army base), which he described as a place he could hardly wait to escape. He came to Berlin for his university education, majoring in computer science at the Technical University. After graduating with a master's degree, he got a job at a small firm in Potsdam, where he met Mihir, who was hired on there as a temporary coder.

We took our seats in a spacious fourth-floor conference room with views of the shopping arcades below and examine our programs, awaiting the first speaker, a representative of Daimler. During the conference, the tenor of discourse around India and Indian IT was a mixture of the practical business strategy to be pursued by outsourcing "lower-level" tasks to Indian IT and the kind of difference and exoticism that could be a supplement to hard-headed business know-how. The first speaker discussed the mutual benefits of doing business with India. The way to approach Indian IT, he said, is to remember that it frees you up to concentrate on the things you are really passionate about and are good at. "Do what you do best," he intoned in English, "outsource the rest," echoing the theme of the CeBIT, or Centrum für Büro und Informationstechnik (Center for Office and Information Technology), expo that had taken place in Hannover a few months before. He claimed that Daimler was doing just that: the German head office provided strategic resources, and its partners in India provided the engineering talent to get the jobs done. In that way, he argued, both countries were winning out. He pointed out how the government's green card initiative was part of the same mentality, bringing Indian programmers to Germany to do the work that is needed by German businesses. They provide the talent; the business provides the leadership.

The next speaker, a manager from SAP, worked out of the company's Bangalore office. His tone was even more reassuring. Bangalore had much to recommend it, but there were three things holding Western companies back from moving all their business operations to the subcontinent: the political turmoil, the unpredictable weather, and the inconsistent access to water and electricity. He thought that German know-how could help the Indians in these areas and, at the same time, assured the German businesspeople in the audience that until such problems were solved—and who, really, can solve the weather?—the engineering talent of India would continue to be paired with German business talent.

These two presentations come to an end, and a hush fell over the crowd. People leaned forward in their seats as a white-shrouded figure with a long black beard entered the room, with a man in saffron strewing flower petals at his feet as he goes. The announcer exclaimed that we were very lucky to have with us today, by very special appearance, the world-renowned Indian guru and founder of the Art of Living Foundation, Sri Sri Ravi Shankar. It was an unexpected turn of events at a meeting otherwise dedicated to extolling the virtues of outsourcing as business process. The crowd of some one hundred men and women in formal business attire seemed relieved by the change in pace. They perked up and gazed at the stage, wondering what the guru might say. The man sitting next to me was explaining to his friend that Sri Sri was very hard to get a hold of, his schedule was very full, and there were hundreds of devotees in Europe alone waiting for an audience with him.

Sri Sri's organization, the Art of Living Foundation, has a divided mandate. It does meditation and empowerment work with prisoners. It also offers seminars and courses for business professionals designed to teach them breathing techniques called *sudarshan kriya*, to combat stress and increase longevity. Sri Sri had prepared a clever, punning message that played on the initialism *IT*. He delivered a talk entitled "IT—Internal Transformation." The audience should all think of IT in terms that go beyond "information" and "technology"; rather, IT could also and should also mean striving for internal transformation. He exhorted everyone in the room to keep this other meaning of IT in mind and "focus not only on the bottom line, but on their internal bottom lines." They should pursue not only success in business but also success in moral achievement. He smiled down on us all as he ended his talk, and the room erupted into enthusiastic applause.

I argue in this book that race in knowledge economies needs to be approached in its duality, as both a regimentation of labor divisions and worker management and an affirmation of liberal selfhood and a fecund source of communicative value. The response that Sri Sri Ravi Shankar's speech received demonstrates the importance of India as a source of value that floats—much like the derivative discussed by LiPuma and Lee—as the singular immaterial commodity of new economic networks.[45] The value of Indian coders can be similarly derived from the value of Indian culture. The former is valued for its cheapness, the latter for its distinction. As Susan Marchand relates in her comprehensive work on German Orientalism, although the relationship between Germany and India was complicated and changed over time, a strong strand of romanticism vis-à-vis Indian spirituality was woven through the history of German encounters with India.[46] It is this discourse that is at work once again in IT offices where points of Indian culture are matters of comment, discussion, and pleasure. As "internal transformation," IT refigures Orientalism so that it can be loosened from its moorings in India and fit within the confines of a business program. No longer strictly separating the spiritual and the material, the conference holds out the promise that the material can be infused with spirituality and that spirituality can be interdigitated with business concerns.[47] The morality of business, encapsulated in the terms that Sri Sri refashions, such as "bottom line" becoming "internal bottom line," is not undermined but preserved.

The conference perfectly encapsulates the two sides of the mobilization of race in cognitive labor. It is used to inaugurate a neat separation between front- and back-office operations and between the creative work that will be done in the first world and the rote work that will be done in India. The guru's speech highlights the *jouissance* that sticks to Indian culture when it is marshaled as communicative event. These two sides of race are not easily reconciled but instead form a constitutive contradiction in the development of capitalism. The confluence of the human and the economic gives pleasure by bringing basic faculties of the human in its communicative capacities in line with the sphere of production. The communicative capacity of race opens up a temporary resolution to the contradiction between working life and life as a value, suggesting that office workers can find satisfaction in their jobs to the extent to which their jobs can be suffused with both technology and transformation, both hard work and spirituality. Of

course, this resolution is only ever temporary; when faced with the pressure of work deadlines, profit margins, the desire to generate leads or new business, and the pressing need to monetize those very communicative, human capacities, the balance between life as valued in its fullness and the value of a productive life begins to fracture.

The image of Indian IT workers that circulates in the offices is multifaceted, while the significance of their difference (from German workers, from other foreign workers) is at once recognized and disavowed as important to the way the office is organized. Indian programmers and programming work, from writing source code to testing existing products, are interconnected through these discourses of performance and exceptionalism. The labor of Indian programmers, as interpreted through repeated moments of noticing and remarking on their foreignness, at once ties them to cognitive economies and separates them out as having a particular role to play in them. The intraction of this human-technological unit proliferates (as Karen Barad and Bruno Latour might write) both kinds of people, as well as pathways for programming technology.[48]

In this chapter's opening vignette, the interaction between Jan the project manager and the office administrator of their firm was about the decal of Lord Venkateshwar in the entranceway. While the administrator, who sits up front and greets visitors, was worried that the manager might disapprove of its presence, Jan simply gave it and her a shrug and moved into the office. Such encounters gesture toward several kinds of inequality. The class difference between administrator and boss might be at issue, with each having a different attitude toward foreignness in the office. The office hierarchies are on notice, with the administrator subordinate to the manager and quick to distance herself from what she believes might be problematic. Jan's indifference signals a kind of acceptance of the supplements that an Indian workforce might bring to the office, and at other times, he is eager to learn what these additions might be. Such additions are tolerated because they do not harm the transaction of business but can enhance it by giving him the information to better manage an Indian workforce and, at the same time, begin to think of India as a new market for development. The sign of Venkateshwar in the entranceway may even signal his office's world-open, global attitude.

Within the official meritocracy of a postracial office, the race of Indian programmers can take on a number of different casts—the safe migrant,

the limited worker, the valuable resource to open new markets and generate new ideas, the necessary supplement to Western materialism, and the alien automaton. In the dream of participation and virtual adventure promised by cognitive economies, the foreign IT worker should add color and promise new horizons of knowledge—and it is the concretization of office hierarchies that will produce this promise as a monetizable bet on the future.

As for Venkateshwar, it turns out that neither of the two programmers currently at Globus put the sticker up. They tell me it was put up by one of their predecessors who has returned to India. Asking them how they feel about it, they tell me it makes them feel good, as Venkateshwar in the entrance will ensure the success of the enterprise. Whether it will ensure their own success, however, remains an open question.