Work and Occupations
37(3) 320–348
© The Author(s) 2010
Reprints and permission: http://www.sagepub.com/journalsPermissions.nav
DOI: 10.1177/0730888410373331
http://wox.sagepub.com
SAGE

The Missing Customer and the Ever-Present Market: Software Developers and the Service Economy

Seán Ó Riain[1]

Abstract

Although some software engineers and developers work directly with the final users of their product to generate customized software, many do not. However, drawing on an ethnographic study of software developers in a U.S. firm in Ireland, this article argues that both software developers who work closely with customers and those who do not can be thought of as "service workers." The article extends the analysis of the "service triangle" of workers, managers, and customers to software workers who interact with customers in the software development and support process. It then uses the case of software workers who do not interact with customers to rethink our definition of what counts as service work. For these workers, the customer also looms large in the workplace—but only as an abstract entity to which they should respond and be attentive, mobilized through organizational mechanisms that transmit and simulate market pressures rather than through concrete interactions with customers themselves. The irony is that an organization of production that mobilizes the customer as the driving force of the production process ultimately, and largely unintentionally, marginalizes the customer as irrational and incompetent—an outsider in the service economy, with little input into the technologies they end up using.

[1]National University of Ireland, Maynooth, Co. Kildare, Ireland

Corresponding Author:
Seán Ó Riain, Department of Sociology, National University of Ireland, Maynooth, Co. Kildare, Ireland
Email: Sean.oriain@nuim.ie

Keywords
software, service work, markets, professional work, knowledge economy

Although some software engineers and developers work directly with the final users of their product to generate customized software, many do not. The sociology of service work has provided important insights into the transformation of work when we add customers to the classic relation between managers and workers. However, whether they work directly with customers or not, software developers work in the service economy. Drawing on an ethnographic study of software developers in a U.S. firm in Ireland, this article examines software development through the lens of the service economy to rethink our understandings of both software work and service work.

This article contributes to the literature on service work in two ways. First, it extends the analysis of the "service triangle" of workers, managers, and customers to software workers who interact with customers in the software development and support process. In this respect, it applies the existing paradigm to an occupation that is rarely conceptualized in that light (but see Barley & Kunda, 2004; Stinchcombe & Heimer, 1988). Second, it uses the case of software workers who do not interact with customers to rethink our definition of what counts as service work. The customer also looms large in the world of software developers who never deal directly with customers themselves. For these workers, the customer also appears as an abstract entity to which they should respond and be attentive—but through organizational mechanisms that transmit and simulate market pressures, not through concrete interactions with customers themselves.

The article asks, for both groups of software workers, what are the processes through which customers, real and abstract, appear in software work. What are the conditions that enable this? How is software work changed by the presence of the customer? To answer these questions requires that we expand the terms on which we have approached the sociology of service work.

Service Work, Service Triangles, and the Service Economy

Korczynski (2009) defines front line service work as "work undertaken where the central job task involves interaction with a service-recipient and where the job status is below that of professional" (p. 952). This definition clearly identifies some central elements of the understanding of service work in sociology, two of which are critical to this article. First, research has focused on workers below the professional level—workers who are relatively weak in the labor market and often disempowered in their relations not only with managers but also with customers. Second, our understanding of the service triangle has been fundamentally interactional—the triangle operates as real relations between actors who meet in the workplace. In particular, analyses of service work focus on relations between the producers of a service (or occasionally a good) and the final users or consumers of that service.[1] At the risk of blurring the focused lens of the sociology of service work, in this article I seek to expand each of these dimensions of Korczynski's definition of front line service work.

The Curious Case of Professionals and Service Work

Korczynski (2009) defines service work as confined to workers below the level of professionals. However, although this serves to focus attention on a clearly defined group of workers, it may in fact make it more difficult to identify which features of the service work experience are linked to the interactional work in the service encounter and which are linked to a more general weakness in the labor market and the workplace. Examining professional workers allows us to examine the varieties of service relationships that exist in the workplace, particularly given that professionals are often dominant partners in such relationships (Abbott, 1988). Indeed, sociologists examining professional work have often been more concerned to protect the "service recipient" from the "service worker" rather than to critique their power as customers.

Most professionals have in one way or another been service providers, often interacting on an ongoing basis with the service recipients. Professional work was an early form of marketized service work, operating typically from a position of high status and often dealing with relatively privileged clients. The development of the welfare state and the corporate economy has shifted these relations somewhat as the typical professional service interaction has shifted from the face-to-face relation to a contracting client to professionals acting in the service of hierarchical employers (public and private) to manage and "serve" that organization's "clients." Furthermore, it appears that "professionalism" as a mode of organization has been losing ground in the face of the commercialization of professional work—transforming the social relations of professional work and professional identities (e.g., Hanlon, 1994, on accountants, and Barley & Kunda, 2004, on software contractors).

Previous research has shed significant light on the forces shaping software developers' work lives. For example, Perlow (1997) finds that long hours are reinforced by the job insecurity of the industry. Workers dedicate themselves intensely to one project so that they will be asked to participate in the next, in what Perlow calls a "vicious work–time cycle." Similarly, Sharone's (2004) study of a Silicon Valley software team shows that the pressure to work long hours and commit to work above all else is reinforced by individualized pay structures. Software developers are ranked in relation to one another, so that their "performance pay" becomes what is in essence "competition pay," driving a competitive pressure to work longer and display a greater commitment than their colleagues.

If labor markets matter, so do the forms of coordination and organization of software work. Perlow (2001) finds significant interfirm and cross-national variability in hours and in time norms in software workplaces in India, China, and Hungary. She links this to the modes of coordination within firms where individualized relationships created a demand for long and overlapping time schedules where more structured team-based patterns of organization allowed for better management of time. Deadlines, driven by crisis management, reinforced the pressures associated with project work (Ó Riain, 2000; Perlow, 1997). Outside the firm, although peers in the software developer occupational community could be a resource in coping with the twin pressures of intense work and job insecurity, the need to maintain reputation among the peer group could also be a further source of pressure on workers (Barley & Kunda, 2004; Osnowitz, 2006).

These pressures seemed less extreme in Ireland—the hours worked appear somewhat fewer than in the United States (although good data are hard to find), and research by Aileen O'Carroll (2004) shows that, in many Irish firms, workers were able to impose a set of time norms of their own, which included a more reasonable set of expectations around working to deadlines and restricting longer hours. Nonetheless, O'Carroll finds that some firms were characterized by exceptionally long hours, partly because the lack of industry norms around working hours allowed for a great deal of variability at the level of the firm. For website production workers, Damarin (2006) finds that "fluid jobs allow workers some autonomy in production, but little control over the wider organization of work" (p. 429)—again allowing for significant pressures and variability, even as workers have significant autonomy in work itself.

Lurking behind these pressures from the labor market and forms of organizational coordination is the ever-present shadow of the market, experienced as an intense set of pressures—even if somewhat indirectly. We know relatively little about the mechanisms through which these market pressures manifest themselves within the software workplace. The shift of professional work more firmly within hierarchies has combined with commercialization of professional work to place professionals more firmly within the triad of worker, manager, and customer that is at the heart of service work. Professionals may experience the same pressures from customers and service interactions as other service workers, despite their greater relative power. This is particularly the case with professions, such as software, that do not exercise the kinds of tight occupational control and closure as the most traditionally high-status professions (such as law and medicine). Such professions have been incorporated within state and corporate hierarchies since early in their occupational histories and have typically exerted much less control over labor supply and other processes of social closure.

Frenkel, Korcyznski, Shire, and Tam (1999) have examined software development as a form of frontline service work. They find significant differences between the work of software developers in "knowledge-intensive" workflows and that of sales or service workers. The software developers are more interdependent with their service recipients, service is negotiated, and there is a significant degree of ambiguity regarding who constitutes the most powerful party in the service relation. Significantly, Frenkel et al. find that although customers can be a source of significant pressure for professional workers, the service triangle is characterized by varying degrees of disconnection among its different elements and shifting and varying alliances between customers, managers, and workers.

Similarly, Barley and Kunda (2004) document the complexities for software contractors of navigating between clients, staffing agencies, and occupational communities. For these workers, clients are a regular and direct presence in their working lives. But in the process of entering the service economy, software developers were pushed to renegotiate the meaning of professionalism itself:

> To close deals contractors also had to engage in complex, threeway bargaining with hiring managers and agents. This process exposed contractors to an unexpected reality. Bargaining was more than just haggling over rates that maximized income. Contractors discovered that they also had to negotiate the very definition of their skills. . . . There was always a gap between what contractors believed they could do, what they said they could do, and what the clients claimed they wanted. To land a job, contractors had to identify and bridge this gap. (Barley & Kunda, 2006, pp. 48-49)

Software developer contractors were defined as much by their service work as by their professionalism—and the two were blended in their everyday lives. In those everyday lives, their often rewarding opportunities were balanced by significant anxiety and insecurity—which were aggravated by an institutional context that assumed firm-based, bureaucratically organized employment.

The implications of Barley and Kunda's (2006) study and Frenkel et al.'s (1999) discussion of software and other professionals in knowledge-intensive work processes is that an analysis of service work remains highly relevant to professional workers. However, we still know relatively little about how such an analysis can inform our understanding of the software work process or about the mechanisms that link these broader conditions of the service economy to work itself. This gap is particularly wide for software developers who work for employers, rather than as contractors. Such an analysis requires an extension of the fundamentally interactional framework through which we have understood service work to date.

Dancing With Triangles: From Customer Interactions to Product Markets

The "service triangle" of customer, worker, and manager has been central to the enlightening analyses of service work that have proliferated in recent years. This has brought us important new insights into the organization of work and the politics of new forms of work and has sharpened our focus on aspects of work that had previously been neglected—including emotions (Hochschild, 1983; Lopez, 2006) and the body (Lan, 2001). However, although interaction with service recipients appears to be increasing, it is neither universal nor necessarily synonymous with the service economy.

The "service work triangle" is a significant element of the service economy but does not fully describe its social relations. The significance of the customer's presence in service interactions is that the pressures of product markets are brought directly into the interactional world of the service worker. In the classic dyadic view of the workplace, workers were subject to the pressures of organizational hierarchies and labor markets but remained somewhat insulated from the competitive pressures in product markets, which loomed large for their employers. The "service triangle" focuses on interaction with customers as the primary mechanism through which product market pressures are introduced into the workplace—it connects service work and the service economy.

However, a focus on “service triangles” neglects the diverse range of mechanisms that can transmit product market pressures into the workplace. For some, all workers and workplaces now exist within an economy where “meeting the demands of the ‘sovereign’ consumer becomes the new and overriding institutional imperative” (Keat & Abercrombie, 1991, p. 3). A “culture of the customer” plays a critical role in incorporating workers into a market society (DuGay & Salaman, 1992). For DuGay and Salaman (1992), the macro-level trends toward marketization are linked to internal organizational processes through a discourse of enterprise within which “the customer” plays a critical role in linking “external” and “internal” processes (p. 617). In this view, increased awareness of customers and other “service recipients” and the rhetorical mobilization of the “sovereign consumer” bring the dynamic of service work triangles into multiple different kinds of workplaces. However, this approach substitutes a totalizing macro-level view for the interactional level of analysis, weakening our ability to analyze the diversity of ways in which service work and service economy are connected.

More promising is Fuller and Smith’s (1991) analysis of managerial use of customer feedback to manage employees—what they call “management by customers” or “consumer control.” The incorporation of customer feedback (through comment cards, evaluations, etc.) becomes an important tool for managers in redesigning work processes and in controlling individual employees. Crucially, for our purposes, this work identifies one of the mechanisms through which organizations can mobilize the “service economy” within the workplace.

Similarly, one of the goals of this article is to explore how software work is shaped by its location within a service economy, without sacrificing the rich analysis of concrete organizational and social processes that has characterized studies of service work to date. The move to the macro level of analysis opens up the analysis of work in the service economy and not simply the analysis of service work itself (Glucksmann, 2009; Sallaz, 2002). However, in that move, the specificity, empirical richness, and groundedness of the sociology of interactional service work can be lost.

The analysis of organizational processes is crucial to avoiding this, and this has directed analysts to pay increased attention to the place of service work within the “hegemonic regime” of the workplace (e.g., Sallaz’s, 2002, study of casino dealers, and Sherman’s, 2007, study of luxury hotel workers) or the broader social formations of labor (Glucksmann, 2009). Lopez (2006) argues that the relations of service work are profoundly influenced by the organization of the workplace itself. In his study of a nursing home, Lopez (2006) finds “an organization that self-consciously tried to create structural opportunities for meaningful social relationships between caregivers and clients" and where there was "organizational support for ongoing human relationships in which the emotional rules can be renegotiated by the participants" (p. 134). Such a perspective integrates the insight that managers and organizations profoundly shape how "the customer" enters the service triangle and the perspective that emotional and service work can take both troubling and potentially enriching forms.

Similarly, I conceptualize the service work triangle as just one element of a broader phenomenon of the increasing interpenetration of organizational hierarchies (and markets for governance), labor markets, and product markets. In addition to the extension of the dominant "service triangle" framework for analyzing service work to software developers who interact with customers, this focus on the broader organization of the service economy allows us to examine service work in the absence of the customer.

Research Method

To examine these questions, I return to an ethnography of a software development team undertaken in 1997 in Ireland, as the high-tech boom was well under way (Ó Riain, 2000, 2004). To investigate work organization I undertook an ethnographic case study in early 1997 of USTech.[2] The study lasted 3 months and was carried out with the permission and assistance of company management and the full knowledge of company employees. USTech was well established in Ireland, having located there in the 1970s and becoming one of the early success stories of Irish industrial policy. For many years, it was one of Ireland's primary computer hardware production operations, with a reputation for high quality. The hardware manufacturing operations of USTech Ireland were dismantled with massive layoffs in the late 1980s and early 1990s, leaving local management scrambling for the operation's survival and turning to a complete reliance on the local pool of software skills.

The case study included 12 weeks ethnographic research on a software team, 20 interviews carried out with engineers and developers working in the company, 15 interviews with company managers and attendance at 15 team, department, and management meetings (plus team meetings of the team in which I worked). For the analysis of software work where customers were present, I rely primarily on interviews with managers and software developers and team meetings of two parts of the company—software testing and support for a product called TPS and applications development and consulting around a technological system called ObjectWorld. These two parts of the company are of particular interest as they involved more direct relations with customers than the software development team—dealing with customer difficulties with the product in the case of testing and support and carrying out new systems development in close contact with the customer (and often at the customer's facility) in the case of applications development.

My research on noninteractive software work was more detailed as I spent 12 weeks working on a software development team as a technical writer compiling a manual and online help text. The five long-term members of the team were employed by USTech but were working on a contract designing a product for Womble Software, a spin-off from USTech headquarters in the Unite States (we were known as the "Womble team"). Womble were designing a system for the networked delivery of training videos and other content that was designed to allow users to manage the pace and content of their own training programs.

In addition to the team leader, Seamus, the Womble team consisted of two permanent (Dan and Conor) and two long-term contract (Jim and Paul) staff. During this time, I participated fully in the work of the team and wrote a user guide for our product, which was installed on the system as online help for users of the system. I sat in the same cubicle as the rest of the team, attended team meetings, and interacted closely with them on a regular basis on decisions regarding the user guide.

I became an accepted member of the team, although my researcher status was never forgotten. Working on the product itself and proving myself of some use was essential to this acceptance, which itself was helped by the flow of contract personnel and others through the team on shorter assignments than my own. Furthermore, I was the person on the team who dealt most directly with the "user" side of the product—carrying out rough testing by attempting to use prototypes of the system, writing help text for users, and occasionally making comments regarding design features to the team leader. As we waited for the design team in California to produce "screens" for us that would be the basis for the "front end" of the system (i.e., what users would see), I generated some rough drawings of the screens of the system we were designing, illustrating the logical flow of each screen. These became the rough template for the team's development work. A critical moment in my acceptance in to the team came when I identified a logical flaw in the path that the user followed through the system, based on my rough sketches. Gingerly pointing this out to the team leader, I was relieved a few minutes later when he turned to me and said, "Good catch, Seán." The moment was not lost on the developers in the cubicle—I had cleared the competence bar (set pretty low for visiting sociologists) and proved myself as a member of the team.

Organizational Conditions of the Service Economy: The Blurring of Markets and Hierarchies

Customers bring pressures with them because they transmit the demands of product market competition into the work process. However, these pressures from product markets may also be transmitted into the worlds of different workers in other ways. This section outlines a variety of mechanisms through which software developers' work is shaped by product markets and their interaction with the more widely recognized influences of organizational hierarchies and labor markets on the organization of work. It identifies these general organizational conditions and then goes on to explore how these play out in contexts where customers are present and in those where they are not.

The dominant organizational forms in the information technology industry have shifted from the hierarchical bureaucracies of IBM and Digital, with strong guarantees of long-term employment and corporate cultures (Kunda, 1992), to the networked system of open innovation, where increasing numbers of technologists work in smaller firms (Mowery, 2009) and have careers that stretch across a number of firms (Brown, Haltiwanger, & Lane, 2006). However, hierarchies have not disappeared. In effect, market relations have developed within and across hierarchical boundaries, blurring the lines between hierarchy and market. In this section, we explore how these processes work through the markets for capital and labor to transmit product market pressures more directly into the software work process.

The Market for Capital

Earlier we emphasized the importance of the incorporation of professionals within managerial hierarchies. However, the primary mechanism for transmitting the pressures of the market and the customer to software workers is the marketization of organizational structures themselves. One of the most significant organizational transformations of recent decades has been the blending of markets with hierarchies, placing units within the same organization in competition with each other and making them responsible for their own survival through contracts with internal and external customers.

USTech had gone through a particularly significant transformation just prior to the time I spent there. Once a company on the model of IBM, the new USTech had moved firmly to the more marketized structure. The effect on USTech Ireland was substantial—as one manager put it:

> The goal up to now has been to support USTech as a corporation getting to customers. Now we are starting to think about what can we provide to third parties? It's part of keeping us secure, not having all our eggs in the USTech basket.

Or, as the financial director put it,

> In the old days, US HQ was 90% of USTech Ireland business. Nowadays, we have a range of customers. In the old days we could rely on Big Brother in the US. The budget these days has to be spoken for by 15 to 20 major sponsors.

Other departments within the company itself (including within the Irish facility) were increasingly thought of as "internal customers." The information systems manager for USTech Ireland spoke of the issues in dealing with other departments in the Irish division in the following terms: "At USTech Ireland the users are professionals and are harder to control. . . . We had one guy started today and half his role is planning customer [i.e., Departments at USTech Ireland] relations."

USTech has therefore gone from a hierarchy within markets to a firm that increasingly consisted of markets within a hierarchy, or certainly a hierarchy that has blended market forms with the existing hierarchical structure. In particular the subunits of the firm are increasingly treated as units competing in markets with very little central funding or services provided to them. In the process, "intermediary customers" between the producer and the ultimate user have become significant actors in the work process. The organizational buffers between workers and markets have been weakened—but largely through the growth of intermediary customers, rather than through the disappearance of organizational structures.

So too have the financial and budgetary buffers. As the USTech Ireland Managing Director notes, "I don't have a budget, it's a customer style relationship within the company and with clients. This has a big impact in that we only provide services that we get paid for." The Financial Director notes the difficulties that this raises for collective shared resources that are used across groups, projects, and different clients (e.g., basic research, HR, information systems).

> We have to be careful that the market will stand what the inter-group costs are. We work back from what the customer in the field will pay. If we grow by another hundred staff we'll be able to spread fixed costs across a broader base, we need the critical mass.

Management of these intermediary relationships is a delicate task. USTech Ireland has to manage relations with both its customers dotted across Europe and with the central USTech offices in Silicon Valley—where some other sections of USTech may in practice be competing for business from USTech Ireland's customers. Elsewhere, a team worked closely with another group in Europe while at the same time seeking to use that support to develop to the point where it could compete with them. The lines between market and hierarchy are significantly blurred.

Customers may have to be discarded or disciplined, but only at some risk to your own group—and even career:

> USTech Ireland relies on service organizations affiliated to other parts of USTech or its partners as their customers in Europe. If those service organizations are doing a bad job we need to let Silicon Valley executives know. It's a delicate job because we still need those organizations around Europe so we can get into those markets. . . . It's very delicate with the service organizations because they could potentially be involved in my appraisal. (Gerry, Operations Manager)

Similarly, obtaining support from within the organizational hierarchy of USTech itself can be critical to market success with customers: "We want a mandate for business process work with ObjectWorld—but it's hard to get customers to tell the corporation that they want that for USTech Ireland unless the corporation is telling customers that's what we can do" (Mike, Consulting Executive). Mike is caught between hierarchy and market—needing the hierarchy to tell customers that his unit can supply the services they need but needing the customers to pressure the corporate hierarchy on behalf of his unit.

Therefore, a critical organizational mechanism in the transmission of market pressure to the world of the worker is the role of intermediary customers and "sponsors" that form a chain from the end-user customer to the worker. Where the firm at USTech had stood firmly between the customer and the worker 10 years before, these pressures were now transmitted to great effect by the formation of units at USTech that had the autonomy to sign contracts but the pressure to support themselves through external customers.

Markets for Labor

The particular character of the labor market in software also generates closer links between worker and product market than in many other industries. This is particularly so when the technological expertise required for the work is both part of the production process, part of the product itself, and indeed part of the process of servicing the product once it is in use. For example, at the time of my research, Java technologies were emerging as crucial to many software systems. But Java and similar technologies are not simply tools or techniques used in producing software but are also part of the product itself.

Workers' careers are tied to the dynamics of product markets in two key ways. First, software developers' careers often involve mobility from producer to client organizations. This is because customers learn a great deal about workers' skills in dealing directly with workers as clients, and the skills of software engineers are carried with them when they move (Stinchcombe & Heimer, 1988). This is particularly relevant for those workers who deal directly with clients. Second, software developers' ability to command a high price in the labor market is tied to the demand for the technologies in which they are expert, creating a link to the product market for even those software workers who do not deal directly with customers.

Mobility between buyer and seller organizations is highly relevant for the consultant software developers who work on site at customers' locations. As the managing director says,

> Trying to get people back from the customer can be hard. The customers want somebody Irish—for example, there are 8 USTech people at Hurtig Insurance in Springfield at the moment. Guys with first class honors degrees from Ireland. . . . They are better than anyone on the US side, the customers love them.

Similarly, engineers working in testing develop both expertise in particular products and ties with the development organizations that would facilitate mobility. The relations are delicate between customers and USTech: "TPS skills are a problem, there are a lot of them in USTech customer sites in Ireland but we're not supposed to hire from them" (Pauline, Support Manager).

But there were issues with bringing consulting staff back from customers too—as customers sought to hire them away from USTech, particularly if the relationship with USTech was not a long-term one. "The customer picks consultants from CVs, that's part of it. They like these people from the beginning [and so they choose them from among USTech consultants]" (Kevin, Consulting Engineer). Again, these tensions are only resolved through a combination of markets and hierarchies—where market mechanisms are dominant the relations tend to become quite unstable:

> We had a contractor in. He had ObjectWorld skills—they are like spoiled children, so few people with the skills, this guy knows he can get a job elsewhere. He wasn't very committed. He went on a customer site, it shouldn't have happened. We don't know what he was up to. (Jean, Consulting Engineer)

In this respect, the creation of market forms within the firm might also dampen interfirm mobility. If a software developer knows that he or she is personally in strong demand, they may be able to capitalize on a strong market position without leaving the firm. Indeed, being part of a firm identified strongly with a "hot technology" such as ObjectWorld may be a valuable resource in bringing your skills to the diverse market for those skills. However, it was unclear if this process was at work in the ObjectWorld team at USTech.[3]

The relation of skills to products was the second dimension that connected labor and product markets. If careers are built on particular skills, these skills in software are partly and importantly linked to particular technological tools and platforms and through this to product markets. The success of particular technologies in their product markets can have significant effects on developers' careers.

Developers maintained their skills through external sources as much as through company training:

> I rely on a few places for information. The vendors of products—Cap Gemini and Siemens, also with other vendors. Training courses, documentation, that kind of thing. (Aisling, Support and Consulting Engineer)

> I get all my information from within the ObjectWorld family as they call it. It is small throughout Europe—it's a list of names, I can email if I need to. There's a bulletin board, I don't really use it. There's a Dutch guy who always knows the answer, they call. There's a danger of becoming an ObjectWorld person, not learning more general skills. (Leslie, Consulting Engineer)

But the labor market can also affect the development of the product market itself: "The big issue for ObjectWorld's success is availability of people—customers may give up on it if there's not enough people to support it or if they are too expensive" (Mike, Consulting Executive). Developers' individual interest in scarcity of their skills is in tension with their interest in the expansion of the market for those skills—which is partly based on them not being excessively scarce or expensive.

In the Womble team, after our deadline passed, workers sought to negotiate a place in forthcoming projects that would provide them with access to skills and knowledge in Java that would set them up well for careers within and outside the firm. This move to Java was critical for the product, although difficult, because as it was a new language, Java development skills were in short supply, and many products did not have Java "drivers" that would enable them to work with a system designed in Java. From the team members' point of view this was a great opportunity—training in Java and experience in developing a complex product in the language would be a huge resource for them in the labor market.

On Ramesh's, the manager of Womble and the chief architect of the Womble Software system, second visit, he treated the whole team to a dinner and a night out on the town. Each one of us, as we sat over dinner and wound our way through the city streets, discussed our future roles with Ramesh—I myself talked over the possibility of doing some further technical writing on a contract basis once my fieldwork was over, Paul discussed his hopes to do some field consulting on the product, Jim and Paul their plans to work on a new technical area of the product, Conor his desire to do work with Java in a particular application of our system. Indeed, we also put in a good word with Ramesh for each other where the different roles seemed complementary. In competition over certain areas, the team members helped each other out in others. But in all cases, workers developed skills with a close eye on developments in product markets with technological tools and platforms as the crucial linking mechanism between labor and product markets.

Through both of these mechanisms therefore, software developers find themselves closely tied to product markets—even where customers do not loom large in their work lives, the demands of the product market still do.

Software Development and the Service Triangle

Software systems never fully match a customer's needs, nor are they completely reliable. Indeed, the "failures" of development of standard software systems are so common that testing, support, and customization are large elements of the software industry. The needs of real customers never completely match the assumptions that development teams made in developing systems. At USTech, there were a number of teams engaged in product testing and/or support for TPS. These teams had a strong development role as they were supposed to respond to issues raised by the TPS development team (located in the United States). However, testing in practice was also driven by crises arising with the product in use—as one member of the testing team put it, "we are driven by the customer demands on us, not by what the technology developer releases to us" (Mark, Test Engineer).

Customers introduced an element of unpredictability into testing and support work because workers in these teams were exposed to the vagaries of the systems in use by customers. As Mark put it, "Something could come in tomorrow and it takes 2 weeks to test something, so it would be drop everything and work on that for 2 weeks. That would involve long hours for us all." A crisis at a customer can have significant knock on effects on workers throughout the software engineering process. This was aggravated when workers were on call for support work, a system that was unpopular with workers, even as they accepted the need to provide such support.

As one former consultant, now working in testing, put it:

> In my first job here I worked long hours. It was chaotic, we were reacting to disasters, the customers were breathing down your neck. Tight deadlines. It's bad when the customer is right there, you can end up working through the night. Something has to give. The quality of the software goes. It just made support horrendous afterwards. (Aoife, former consultant)

The customer's desire to "get the job done" can exceed even the developer's, ultimately weakening the quality of the work done and externalizing the costs of this development process into lower status jobs within software such as testing, support, and debugging.

Workers sought to manage their customers to generate a degree of predictability. Test engineers sought to control the pressures on them by coming up with better work processes. These include better monitoring and systematization of the testing and support process:

> There is bad version control on that product. Marketing gives customers anything they ask for without thinking about it. We can never tell what versions customers have of things, it's hard to deal with problems then. The last 6 months here it's been "how the hell are we going to get this done in the time." Have never had a chance to do any automation, no time in last 6 months. The marketing guy in the US doesn't have a clue what's involved. (Mark, Test Engineer)

For these software developers and test engineers—at least in a booming labor market—routinization and automation provide a way for them to manage uncertainty in their environment, as much as for management to exert control.

The attempts to control these pressures can also be more direct:

> There's lots of energy, but that can get drained out of you. If you can, you need to make a valid case—try and set longer deadlines. Support needs an immediate response—you have to stay late on just one thing. It's the nature of the business. But it is also the nature of people—you can say no, and you will eventually. (Aisling, Support and Consulting Engineer)

Teams communicate with other teams in USTech that have relevant skills and information, even if they are not directly involved in the process—the test team may recreate a technical situation to assist the support team in solving a problem, for example. Networks across teams and managers can also be used to attempt to control the flow of such demands:

> Our manager is aware of important issues, the hot issues from customer complaints and so on. The manager in TPS Test section keeps us informed of changes, medium-term decisions and important short-term decisions. (Aidan, Test Engineer)

Another section of USTech provided consultants to work directly on corporate customers' systems that used a specific set of technologies, labeled ObjectWorld. The ObjectWorld consultants were able to do some of their work from the USTech facility but also spent a great deal of time at the customers' facilities. When consultants had to leave home to work on-site at the customers', work hours increased enormously as the direct pressure from customers to get the task completed in the number of days allotted was intense—and often directly enforced by customers.

One consultant engineer said that her work hours averaged 40 to 50 hours while at USTech but stretched to 50 to 70 hours when on site with a customer. On-site work brings a more intense exposure to the market in two ways. Working directly with customers places the kind of pressures of immediacy and monitoring on workers that we normally associate with the "service triangle"—not surprisingly, given that this work situation is the closest software engineers come to that kind of service relation. In addition, workers are also geographically isolated from their home and nonwork life. They often socialize with workers at the customer site, and the customer is generally aware that the visiting engineers have weaker claims than other workers regarding family or other nonwork obligations. Within the consulting teams, these pressures are explicitly recognized: “Everyone knows what it’s like to be out on the customer site so if someone calls with a problem from a site people drop everything” (Leslie, Consultant).

Therefore, there is a group of software workers whose working lives are profoundly shaped by interactions with customers—in the “service triangle.” They are generally hired from the producing firm but are controlled largely by the client. They strategize much as other workers do to manage the demands of customers and seek to mediate client influence through a variety of alliances with fellow workers and even occasionally managers and clients.

The Missing Customer

What then of the software developers who work in isolation from customers? When workers are developing standardized software products, or to a set specification, they typically work in isolation from specific customer needs and demands. Software developers, given the difficulty of separating design and development, regularly make decisions with implications for users even when their work does not bring them into direct contact with customers. Nonetheless, most such software developers work as a team, in relative isolation from the end users of their products and systems and often even isolated from the primary customers (who are often corporate entities).

There was a pervasive sense at USTech of pressure of time and external competition, perhaps not very surprising in a company where many manufacturing workers and engineers had been laid off in the previous year and many new employees hired into the software development area. However, we worked in isolation from these pressures on an everyday basis. We had few dealings with customers—either corporate or end-user. As developers, we heard little from the external manager of Womble Software, located in the United States, although he spoke with the team leader every day and made a long-anticipated visit to USTech Ireland shortly before our deadline for delivering the system. Nonetheless, although the market was ever-present at USTech, the customer was conspicuous by his or her absence.

This was not because the users of the product were irrelevant. Indeed, there were a variety of potential customers for the Womble team. First, there was the potential influence of a generalized sense of competition in the market for training software. But there were more concrete potential customers also. The second potential customer was, most obviously, the final end user of the product—the person who would some day sit at his or her computer and use the system to steer through a customized, self-paced training program. Third, there were the corporate managers of the organizations who would employ these staff—the Chief Information Officers and Human Resource Managers who might be involved in the purchasing decisions around Womble Software. Fourth, there was the sales and marketing division of USTech and Womble itself, which was to connect Womble to customers. And, finally, there was the Womble organization itself, both manager of the Womble team from a distance and contractor for its services. How did these potential customers affect the work of the Womble team?

First, perhaps the abstract form of "the consumer" was powerful enough to generate the pressures of the service economy within the team (Gamble, 2010; Korczynski, 2004)? There were certainly attempts at USTech to achieve this through the rhetorical mobilization and valuing of the customer (DuGay & Salaman, 1992). New recognition or reward schemes were introduced in the company, including small rewards for referring customers (Fuller & Smith, 1991). Existing quality enhancement schemes were redesigned, adding in business results and not just quality in the production process itself. As the quality manager noted, "This is a new focus . . . more of a customer focus." This manager noted that a major issue for his group were concerns about how to measure customer satisfaction in order to incorporate it into quality assessment.

There was also significant awareness of competition from other firms and other countries among developers themselves—and not just among the managers with the main responsibility for facing that competition. The managing director talked at length about being close to the customer and having to seek out a variety of customers, in contrast to when they operated as a production centre for the main corporate headquarters in the United States. For the developers, occasional references to Indian software developers evidenced both respect for their skills and an awareness of competition with them (Ó Riain, 2000).

Despite this rhetorical emphasis on customers and the market, there were clear limits to these strategies. Developers were skeptical of any initiatives that emphasized corporate culture or a commitment to the interests of the corporation—and quality enhancement and customer satisfaction fell well within those boundaries.

Second, what of the final user of the system, who would actually interact with the technology? For those developing the system, the user was a distant concern. Ramesh had written a "white paper" that outlined the educational and learning philosophy underpinning the product and the relevance of the system to the user—but it remained unread and unnoticed. More significantly, the user rarely figured in discussions of the details of technical development. Whereas some of those technical decisions might have little relevance for end-users, others had implications for the logical flow through the system or the kinds of data that users could access and the points, sequences, and formats in which they could be accessed. As we discussed the flow of commands that would take users through the system, I asked "would that make sense to a user though?" Conor sardonically commented, "User? There's a user?" But for the historical accident of my arrival as a researcher, no one in the team would have explicitly concerned themselves with the user's interaction with the system. As Conor commented with a rueful shake of his head, "customers, God save us from customers."

Third, although the purchasing decisions of corporate information officers presumably loomed large for Ramesh, they were never remarked on in my 3 months at Womble.

Fourth, and similarly, the Womble team showed disdain for customer-related aspects of the corporation. Developers criticized the skills and contributions of sales and marketing workers, particularly when compared with technical workers. In contrast to their respectful tone with regard to Indian software developers, they commented disparagingly on the "American" sales and marketing focus. Their critical views were only confirmed when a visiting marketing executive braved the team cubicle to try to access his external e-mail account. His failure to get online to get his e-mail rapidly turned into a widely enjoyed story over the following days. Marketing was conspicuous by its absence from the cubicle and the work process.

The irrelevance of the user and customer in the development process suggests that neither interaction nor rhetoric could bring the customer and market into the development process. This leaves us with a puzzle—if customer interaction, market rhetoric, and customer feedback had little effect on the work process, what organizational mechanisms did bring the market into software development?

The Ever-Present Market

Ironically, it was Ramesh, the manager of Womble, who played the most central role in bringing in the pressures of the product market. It was Ramesh who was keenly aware of the pressures of market competition on Womble and of the purchasing decisions of corporate information officers. He had written the paper that located the design of the system in relation to a philosophy of training and learning. It was Ramesh who worked closely with the sales and marketing divisions. These pressures were funneled through Ramesh into the team. Once again, it is not the triumph of the market that brings the service economy into software development—it is the blurring of market and hierarchy. But what were the mechanisms through which Ramesh channeled these market pressures into the team?

In the classic definition of service work, the producer and consumer are present together, and the service is typically consumed in the same time and space in which it is produced (e.g., haircuts). There are three key elements to such a service relation that generate pressures on service workers. Service recipients' needs and demands are presented as immediate and urgent, within the norms of service interaction as much as by any specific actions of the customer—workers know that customers will expect their immediate attention or they are likely to take offence. Service recipients can also define the content of their needs and demands on the spot—perhaps, asking for more work to be done on their haircut, for a sandwich to be heated, and so on. Finally, the monitoring of workers is intensified by the presence of the customer and directs our attention to the control of the body and the emotions that has been so central to the study of service work.

In software development, such immediate monitoring by customers is rarely present. Indeed, the ability to monitor software development is in general quite limited, with managers and colleagues struggling to assess worker effort and productivity on an ongoing basis.

The fact that the team's direct manager, Ramesh, was located some 5,000 miles away allowed the team, including the team leader, to screen information from him in order to let the team balance the technical and time demands to their own satisfaction. Having encountered a particularly thorny problem, the team finally found a solution:

> Jim: So we're going to do that then. Ramesh never needs to know about it. So we can have it set up the way we want it and he'll have it the way he wants too.
> Paul: So we're going to do it the sneaky bastard way
> Séamus: I like the sneaky bastard way!
> Paul: And Ramesh never needs to know
> Séamus: No, no. Well done gentlemen!
> Jim: Just don't say anything about this on Monday when Ramesh is here!

As Conor advised me when I had sent an e-mail to Ramesh about a problem in the "help" screens:

> Be careful what you send to Ramesh. Cc it to Séamus or better yet send it to Séamus first, let him decide. That's what I do. You have to look after your own behind first you know. I try to get involved as little as possible

with Silicon Valley, I give it to Séamus. That way I have a buffer between me and the US.

Therefore, Ramesh had a very limited ability to bring market pressures into the team through the kinds of direct client control that customers were able to exercise over the on-site ObjectWorld developers or the beleaguered TPS test engineers. However, both the time pressures and the content of what "the market" "demands" are very significant in software work, creating two channels through which product markets shape software development work.

Market Time: Deadlines and Product Markets at Work

The "demands of the market," ever present as an abstract category, are made real through two organizational mechanisms, the first of which is the project deadline. As "time to market" and "first mover advantages" become more critical to competition in technology industries—and are believed by managers to be more critical—these market pressures are made real through deadlines for systems development. These deadlines are largely prospective and based on estimations and expectations rather than on direct competition with other firms' internal development deadlines. The timeline for new products entering the market is fundamentally uncertain, adding to both the contingency of deadline setting and the pressure to make those deadlines as short as possible. As such, deadlines do not just "reflect" the market. It is up for negotiation what these market competition pressures mean and what specific dates they become attached to. Deadlines were set through negotiation, but the major source of pressure around the deadline came from Ramesh.

The mechanism for controlling the software development team is the project deadline. As it is impossible for the final design specifications to provide solutions to every issue faced by the team and the actual work done by the team is difficult for management to supervise directly, the deadline becomes the focus of management and team efforts. "Do what needs to be done to get this specification working by the deadline" is the broad task of the team. The deadline is the mechanism by which management brings the time pressures of product markets into the heart of the team.

In the weeks before 1 March, the release date for our product, life in the Womble cubicle becomes busier and busier. The team works longer hours and becomes more and more isolated from the life of the company around them. Internally, the team becomes more cohesive, communication becomes more urgent, technical arguments take on a new edge, and any delay or new instruction from outside the team is met with a barrage of criticism.

The time allotted for particular development tasks is counted in weeks and then in days. Although not as long as the hours worked by some other software development firms in Ireland, the work hours do start to creep up toward 60 a week. Séamus, the team leader, works constantly, often late into the evening and the night.

Weeks earlier, Conor had told me:

> I've a feeling this is the calm before the storm. My attitude when its calm is get out of here at 4 or 5 cos when it gets busy. . . . You have to draw the line yourself as far as hours go, you have to say once in a while "sorry I have something on tonight, I can't stay." You have to keep your standard hours around 39/40. If you let your standard hours go up to 45 then they'll still come to you and ask you to do a few extra hours that evening, they won't think about that extra 6 hours you're doing as part of your standard. It's up to yourself to draw the line.

As the deadline nears, however, Conor ends up staying late and coming in two weekends in a row. The pressures of the product market are not only transmitted to the team but take their toll, much as interaction with service recipients can take a different kind of bodily and emotional toll on other service workers.

Most striking perhaps is that the deadline takes on a "taken for granted" quality. Although deadlines do slip in the software industry, firms seek to avoid this where possible—understandably, given how central the deadline is to the organization of work and managerial control. The Womble software developers do not seek to change the deadline—in part this is because extending the deadline only extends the period of direct pressure from "the market."

Negotiating With the Abstract Customer: Technical Specifications and the Content of Work

But the timelines and rhythms of "the market" are only part of the story. The content of what gets done by the deadline—what the market (customer) "wants"—is also up for grabs. As one developer recalled from a former workplace: "They said 'come and go whenever you want, just deliver' . . . but you should have seen what they asked us to deliver . . ."

The content of the project work to be done by the deadline was contained in the product technical specification (PTS), a detailed document outlining the technical basis and logic of the system and supposedly defining the key aspects of the actual development process. However, in contrast to the expectations of formal models of software engineering, the specification document was necessarily vague in places and could not capture all the technical dilemmas that arose during the development process.

Beyond the general design and specific technical interfaces and requirements of the system, the PTS rarely figured in the work of the development team. Although debates among members of the team were both regular and passionate, I never heard the PTS referred to in these arguments, let alone used in an attempt to trump someone else's argument.

Nonetheless, the PTS could serve strategic uses. Dan, sitting beside me, constantly justified his resistance to certain new tasks that arrived in before the deadline with the refrain "if it's not in the specs, I'm not doing it." On one occasion, Jim and Paul discussed a new requirement for the system that had come in from Ramesh in an e-mail that morning:

Jim: Is it in the specs?
Paul: No.
Jim: Well screw it then, we don't need to do it.

However, they later came up with a solution to the problem, which they knew was not strictly compatible with the technical requirements of the PTS but would solve the problem satisfactorily. In this case, they were willing to drop their apparent dedication to following the specs in order to try to slip a different solution past Ramesh:

Paul: I have a feeling we're going to get f#*!ed on this. I think the thing to do is to keep our mouths shut, do this what I'm doing now, present it to them without saying anything and then if they come back saying "we're not supporting that" then OK. Cos if I just say it to him, he'll just say "Noooo . . . ."
Jim: Yeah, he does that.

It is noteworthy here that the PTS was ignored to produce a "better" solution rather than simply to do less for the deadline. In general, team members were careful to protect themselves from undue interference from HQ in the United States and left the negotiation of deadlines and larger technical issues to Séamus, the team leader. Séamus's discussions with Ramesh at 4 p.m. every afternoon were listened to carefully by the eavesdropping developers in the shared cubicle—when Séamus declared in frustration one day, "just because you say it can be done Ramesh doesn't mean it can be done," it rapidly became a catchphrase within the team. Where the deadline remained fixed, the negotiation of tasks to be completed by the deadline was never fully resolved.

Indeed, elements could slip under pressure. From time to time, a particular problem was put aside for the 10 March release, which was to contain the fixes for the initial bugs (errors) in the system, creating some dissatisfaction among the developers:

> We're all tired, we've been at it for 2 months really. It's a lot of pressure. Something every day. There's no time to take a day and research something. We need a week to go over some of the bigger issues, have some meetings, go over things, you know. There's some dodgy code in there too. (Conor)

Furthermore, the teams were careful in their release of information to managers—often seeking to manage the flow of tasks being given to the team before the deadline. On one occasion, Ramesh sent an e-mail about a "work around" the team would have to do around a problem in the database they were using. Not realizing that Dan had been working on this issue for a while now, he set aside a day the week before the release for Dan to work on it.

> Jim: Dan will have that done today.
> Sean: So what about the day Ramesh is setting aside for it next week?
> Jim: Oh God, I'm not going to tell him we already have a solution. He's already expecting it to slip a bit so if we get it in on time he'll be really happy. I think we're a little bit ahead of schedule but he thinks we're a bit behind so that suits us.

Even the largely absent and imagined user of the Womble software was not spared the ire of the Womble development team. As the team turned in on itself, team members emphasized how irrational and stupid those outside the team were. Technical issues were dominant, whereas questions of usability or how users might use the system in their specific organizational contexts were largely absent. In the brief discussions that did occur regarding how users might interpret certain sequences of commands or ways of navigating the system, developers typically emphasized how users were prone to misunderstand technical systems rather than identifying problems in the design of the system itself. The isolation of the team and the valuing of its perspective on the development of the product reinforced this.

Even in narrow business terms, this is less than ideal. As one experienced project manager noted,

> Men are much better behaved if there's women around! For sure, in every case! It's a better team if there's women in it. Because they're better behaved. They act more mature, it's easier to deal with scenarios. Not twice as mature, it's a marginal thing. . . . I've never been on a team with a vast majority of women. On a back-office team, all men—you get the more boisterous, dressing room atmosphere. And you can't just turn it off when the customer appears. It's a more cordial atmosphere with women there, you don't need to turn it off.

Even leaving aside the gender implications of this observation, the connection between the isolation of the "back office team" and the inability to deal with (even corporate) customers is clear. The Womble team was certainly characterized by this boisterous atmosphere, to the extent that one senior manager expressed his disappointment at the team's inability to present an appropriately professional image to a visiting marketing executive.

Negotiating the timing and content of the work to be done in the software development team essentially involved negotiating with the market for that software. However, this happened at a distance from the market and in ways that were mediated through managers and the legitimated institutions of the deadline and the technical specification. These pressures interacted with techie culture and team processes to reinforce the exclusion of the concerns of the user from the development process, even as they reinforced the pressures of the market.

Conclusion

Customers and service triangles can be important sources of pressure for software workers. However, what we have seen is that the service economy can manifest itself in software work in various ways—through the flow of customer problems and requests, interactions with customers themselves, and deadlines and project specifications,. However, in each case, it is the combination of integration into product market dynamics with pressures from managers and others within the employing organization that generates the specific kinds of pressure that affect these different groups of software workers. Although studies of service work may appear to have focused primarily on the addition of "the customer" to the manager–worker relation, and thus directed our attention to customers themselves, the substance of these analyses in fact emphasizes the interactions among all parties within the triangle.

The irony is that an organization of production that mobilizes the customer as the driving force of the production process ultimately, and largely unintentionally, marginalizes the customer as irrational and incompetent—an outsider in the service economy, with little input into the technologies they end up using. It is the combination of customers and managers that makes the triangle a source of pressure for workers and software developers prove to be no exception—whether they meet customers in their everyday work or not.

Acknowledgments

Many thanks to Dan Cornfield, Steve Lopez, Art Stinchcombe, and Chris Tilly for exceptionally helpful comments.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the authorship and/or publication of this article.

Funding

The research upon which this article is based was funded by the Social Science Research Council and Andrew Mellon Foundation.

Notes

1. There is a third dimension. As a consequence of its focus on the interactional pressures faced by relatively powerless workers, the sociology of service work has focused primarily on the critique of the demands of the service encounter and the costs in terms of emotion, control, and authenticity. New work in the area has shown that, under certain conditions, relations between workers and "customers" can be structured in more rewarding and empowering ways (Lopez, 2006). This is relevant in software also, where movements such as "open source" development have reconstructed relations among users and producers, blurring the boundaries between those categories. However, that issue lies beyond the scope of this article.
2. All names of organizations and persons in the article are pseudonyms. The names of a number of products and technologies have been changed, except in cases where they were in very widespread use within the industry. Some details regarding the companies discussed in the article, USTech and Womble Software, have been changed slightly, although not in ways that affect the substance of the analysis.
3. I am indebted to Art Stinchcombe for this point.

References

Abbott, A. (1988). The system of professions: An essay on the division of expert labor. Chicago, IL: University of Chicago Press.

Barley, S. R., & Kunda, G. (2004). Gurus, hired guns, and warm bodies: Itinerant experts in a knowledge economy. Princeton, NJ: Princeton University Press.

Barley, S. R., & Kunda, G. (2006). Contracting: A new form of professional practice. Academy of Management □erspectives, □□, 1-19.

Brown, C., Haltiwanger, J., & Lane, J. (2006). □conomic turbulence: Is a volatile economy good for America. Chicago, IL: University of Chicago Press.

Damarin, A. (2006). Rethinking occupational structure: The case of web site production work. □ork and □ccupations, □□, 429-463.

Du Gay, E., & Salaman, G. (1992). The culture of the customer. □ournal of Manage□ment □tudies, □□, 615-633.

Frenkel, S., Korcyznski, M., Shire, K., & Tam, M. (1999). □n the front line: □rgani□□ation of work in the information economy. Ithaca, NY: Cornell University Press.

Fuller, L., & Smith, V. (1991). Consumer reports: Management by customers in a changing economy. □ork, □mployment □ □ociety, □, 1-16.

Gamble, J. (2007). The rhetoric of the consumer and customer control in China. □ork, □mployment □ □ociety, □□, 7-25.

Glucksmann, M. A. (2009). Formations, connections and divisions of labor. □ociol□ogy, □□, 878-895.

Hanlon, G. (1994). The commercialisation of Accountancy: □lexible accumulation and the transformation of the service class. Basingstoke, England: Macmillan.

Hochschild, A. (1983). The managed heart. Berkeley: University of California Press.

Keat, R., & Abercrombie, N. (Eds.). (1991). □nterprise culture. London, England: Routledge.

Korczynski, M. (2004). Back-office service work: Bureaucracy challenged? □ork, □mployment □ □ociety, □□, 97-114.

Korczynski, M. (2009). The mystery customer: Continuing absences in the sociology of service work. □ociology, □□, 952-967.

Kunda, G. (1992). □ngineering culture: □ontrol and commitment in a high□tech cor□poration. Philadelphia, PA: Temple University Press.

Lan, P. C. (2001). The body as a contested terrain for labor control: Cosmetics retailers in department stores and direct selling. In R. Baldoz, C. Koeber, & P. Kraft (Eds.), The critical study of work: □abor, technology, and global production (pp. 83-105). Philadelphia, PA: Temple University Press.

Lopez, S. H. (2006). Emotional labor and organized emotional care: Conceptualizing nursing home care work. □ork and □ccupations, □□, 133-160.

Mowery, D. C. (2009). Plus ca change: Industrial R&D in the third industrial revolution. Industrial and □orporate □hange, □□, 1-50.

O'Carroll, A. (2004). In the shadow of the clock (Unpublished doctoral thesis). Trinity College, Dublin, Ireland.

ÓRiain, S. (2000). Net-working for a living: Irish software developers in the global workplace. In M. Burawoy (Ed.), Global ethnography: □orces, connections, and imaginations in a postmodern world (pp. 175-202). Berkeley: University of California Press.

ÓRiain, S. (2004). The politics of high□tech growth developmental network states in the global economy. New York, NY: Cambridge University Press.

Osnowitz, D. (2006). Occupational networking as normative control: Collegial exchange among contract professionals. □ork and □ccupations, □□, 12-41.

Perlow, L. (1997). □inding time. Ithaca, NY: ILR Press.

Sallaz, J. J. (2002). The house rules: Autonomy and interests among contemporary casino croupiers. □ork and □ccupations, □□, 394-427.

Sharone, O. (2004). Engineering overwork: Bell curve management at a high-tech firm. In C. Epstein & A. Kalleberg (Eds.), □ighting for time (pp. 191-218). New York, NY: Russell Sage.

Sherman, R. (2007). □lass acts: □ervice and ine□uality in luxury hotels. Berkeley: University of California Press.

Stinchcombe, A., & Heimer, C. (1988). Interorganizational relations and careers in computer software firms. □esearch in the □ociology of □ork, □, 179-204.

Bio

Seán Ó Riain is a professor of sociology at the National University of Ireland, Maynooth. His research examines the politics of the information economy through studies of developmental network states, regional and national development strategies, and the politics of high-tech workplaces. He is the author of The □olitics of □igh Tech Growth: □evelopmental □etwork □tates in the Global □conomy (2004). Current projects include studies of the dynamics and politics of the Silicon Valley-Ireland production system (with C. Benner) and a life history study of social change in 20th-century Ireland (with J. Gray and A. O'Carroll).