Organization Science
Vol. 22, No. 1, January–February 2011, pp. 262–285
ISSN 1047-7039 | EISSN 1526-5455 | 11 | 2201 | 0262
DOI 10.1287/orsc.1090.0511

Teaching-Learning Ecologies: Mapping the Environment to Structure Through Action

Diane E. Bailey
School of Information, University of Texas at Austin, Austin, Texas 78701, debailey@ischool.utexas.edu

Stephen R. Barley
Center for Work, Technology and Organization, Management Science and Engineering, Stanford University, Stanford, California 94305, sbarley@stanford.edu

Although organizational theorists have long argued that environments shape organizational structures, they have paid little attention to the processes by which the shaping occurs. This paper examines these processes by showing how environments shape teaching and learning activities, which in turn shape structure. Observational field data from structural engineering groups in three firms and hardware engineering groups in three firms revealed that the two occupations exhibited different patterns of learning episodes and different distributions of actors across those episodes, or what, following the work of Roger Barker, we call two distinct teaching-learning ecologies. After detailing the differences in the two ecologies, we show how these differences emerged from patterns of behavior that were influenced by unique sets of environmental and technological constraints. By demonstrating how actions transform environmental constraints into organizational structure, this paper indicates how research on individual learning in organizations can speak to larger concerns in organizational theory. Moreover, by adopting a synthetic and pragmatic approach to individual learning as a social activity, the paper highlights the role of teachers in workplace learning and casts doubts on the existence of a universal model of how individuals learn at work.

Key words: environment; structure; action; learning; teaching; ecology
History: Published online in *Articles in Advance* March 11, 2010.

Introduction

No idea in organization studies is more enduring than the notion that environments shape the structures of organizations. This was the fundamental insight of the contingency theorists, who conceptualized environments using such abstractions as complexity, stability, and predictability, which they applied to a variety of domains ranging from technology to markets (Woodward 1958, Burns and Stalker 1961, Perrow 1967, Lawrence and Lorsch 1967). In general, contingency theorists argued that bureaucratic structures were less appropriate and effective as environments became more complex, dynamic, and unpredictable. Because most contingency theorists paid little attention to the processes by which environments shaped structures, they portrayed the relationship as deterministic. Those who did address process generally confined their analysis to managerial agency, assuming that managers could assess the organization's environment and choose structures that would optimize "fit" between environment and firm (Child 1972, Galbraith 1977).

Later organizational theorists balked at contingency theory's overly rational accounts of how alignment occurred. Population ecologists dispensed with human agency and the idea of alignment in favor of an evolutionary account (Hannan and Freeman 1977, 1988). Neoinstitutionalists retained the idea that people align organizational structures to environments, but conceptualized the environment in terms of values, laws, beliefs and taken-for-granted assumptions rather than abstract attributes of technologies and markets (Zucker 1977, Meyer and Rowan 1977, Meyer et al. 1981). Alignment became a matter of conformity to institutions through coercive, mimetic, and normative processes rather than a matter of optimization (DiMaggio and Powell 1983). Early papers that set out the neoinstitutionalist perspective drew on ethnomethodology (Garfinkel 1967) and the social constructionism of Berger and Luckmann (1967) to argue that institutions were grounded in the everyday behaviors of culturally knowledgeable actors. However, over time, researchers allowed the microsocial foundations of institutional analysis to atrophy and actors to disappear from their research as they turned their attention either to documenting the diffusion of specific organizational structures and practices (Tolbert and Zucker 1983, Dobbin et al. 1993) or to exploring pressures created by centralized authorities and regulatory agencies (Kelly and Dobbin 1999, Edelman and Suchman 1999).

Recently institutionalists have called for a return to the microsocial foundations of institutions, arguing that without focusing on action it is difficult to explain how institutions are created, maintained, or changed (Barley 2008, Powell and Colyvas 2008). Much of this new research has highlighted the contributions of "institutional entrepreneurs," who act as change agents (Lawrence and Philips 2004, Battilana 2006). However, as Powell and Colyvas (2008) have noted, focusing on institutional entrepreneurs overemphasizes the role of "heroes," while failing to appreciate the fact that the everyday activities of all members of a social collective are crucial to the dynamics of institutionalization. The idea that an institution cannot exist unless it is instantiated in everyday life has been a recurrent theme in the study of social organization. It is central to ethnomethodology (Heritage 1993), Berger and Luckmann's (1967) notion of the sedimentation of structure, the Chicago School's research on social worlds (see Barley 2008), Giddens' (1976, 1984) theory of structuration, and Bourdieu's (1977) concept of habitus. Although researchers usually study action's shaping of structure within the boundaries of an organization (Barley 1986, Orlikowski 1992, Feldman and Pentland 2003, Feldman 2004), one can use the same principle to examine how individuals' activities mediate the environment's shaping of organizational structures.

One obstacle that stands in the way of this agenda is the multidimensionality of the environment. Environments exhibit a range of properties, some of which are institutional, such as laws and financial practices, and others of which are not institutional, such as technologies, markets, and bodies of knowledge. For a tractable analysis of how environments affect structures by shaping action, one must narrow the focus to aspects of the environment that serve as constraints and affordances for the specific types of action that the analyst will study.

Similarly, analysts need a conception of structure that can be linked to patterns of action. Organizational theorists have used the term "structure" in many ways. Some take a macrosocial perspective and define structure as global institutional arrangements such as bureaucracy or professional dominance. Others refer to formal attributes of organizations such as span of control or levels of hierarchy. Still others use the term for the social organization of work, which includes the repetitive features of day-to-day activity and the networks inscribed by ongoing interactions. The latter approach seems most useful because it allows analysts to remain close to activity without sacrificing the configurational focus associated with studying formal organizational structures. In fact, most properties of formal structure (for instance hierarchy or differentiation) can be expressed as network configurations (Krackhardt 1994).

Finally, one needs to select a domain of everyday activity for analysis. Among the various activities on which one could focus, learning seems particularly appropriate for three reasons. First, some form of learning occurs in all organizations, even if it involves nothing more than training new employees. Second, organizational theorists have long conceptualized learning as critical to how organizations align themselves with their environment (Burns and Stalker 1961, Kogut and Zander 1992, Adler 1993). Third, network analysts have repeatedly shown that learning, at least when construed as the seeking and giving of information, inscribes networks that reflect and constitute the formal and informal structures of organizations. For instance, Allen (1977), Burt (2005), and others (e.g., Lazega et al. 2006) showed that information flows in organizations demark boundaries among departments and functions. Powell (1990) argued, and Hinds and Kiesler (1995) showed, that information flows also reveal whether work systems are vertically or horizontally structured.

To study learning as a link between environment and structure requires conceptualizing learning as a social activity. Although learning clearly has a strong cognitive component, it is as a social and relational activity that learning contributes to a setting's social organization. Because most learning occurs as part of the round of daily work, researchers also need an approach for separating episodes of learning from an ongoing stream of behavior.

Learning as a Social Activity

Over the years, organizational theorists have approached learning as a social activity from three different perspectives. The first perspective, originally articulated by March and his colleagues (Cohen et al. 1972; March and Olsen 1975, 1976), holds that learning is social because rationality is socially embedded. March challenged the rational models of decision making that dominated managerial theory in the 1950s and 1960s by showing how social incentives, norms, and power affect the way we make sense of what we observe. Although March recognized the importance of social processes in shaping learning, he conceived of those processes as cultural forces that mold what individuals perceive, think, and believe. In this sense learning was social, but not relational, because March's sense makers did not explicitly interact with others as they learned. Accordingly, although March and colleagues wrote about how members of organizations respond to environments, aside from referencing the consequences of managerial decisions, they did not address how learning might transform environmental pressures into structure.

The second perspective on learning as a social activity, associated most closely with Lave and Wenger's (1991) theory of situated learning, brings action and interaction to center stage. Situated-learning theorists rejected goal-oriented notions of learning as well as the idea that learning is separate from an ongoing stream of action. Building on Lave and Wenger (1991) as well as Orr (1996), Brown and Duguid (2001) claimed that learning occurs contextually, subtly, and ubiquitously as individuals participate in a community of practice. To illustrate this argument, Cook and Brown (1999, pp. 395–396) described how apprentices in a group of flutemakers learned to craft quality flutes through repeated interactions with more experienced flutemakers, who returned any flute they deemed "clunky" until it had the "right feel." Beyond making such terse, evaluative comments, the flutemakers never articulated what it took to make a good flute. Situated-learning theorists further held that teaching and learning are often unintentional. Lave and

Wenger drew on Jordan's (1989) study of Yucatec midwives to make this point. Among the Yucatec, midwifery was an occupation passed from generation to generation. With no explicit guidance, girls progressed from watchful children who observed their mothers interacting with expectant families, to assistants who handled small errands, and then to master midwives. Both examples typify situated learning theory's premise that knowledge is passed from old-timers to newcomers in the course of doing. By portraying learning as the informal transmission of often tacit occupational knowledge, situated learning theory's view strongly resembles socialization into a role (Wenger 1998).

Situated-learning theorists construed the learning context as local and composed of artifacts, people, and tasks at hand (Suchman 1987, Lave and Wenger 1991) as well as the understandings (i.e., culture) unique to a community of practice (Brown and Duguid 1991, Sole and Edmondson 2002, Yanow 2003). Largely absent from this concept of context is the external environment in which the community operates. As Gherardi (2006) noted, situated-learning theorists often have drawn examples from preindustrial or artisanal settings rather than from modern organizations and occupations. This focus may partially explain why they have downplayed larger environmental pressures.

Situated-learning theorists have also paid little attention to how communities of practice are structured (Lazega et al. 2006). Rejecting the idea that newcomers progressed from a periphery towards a core of old-timers, Lave and Wenger (1991, p. 36) allowed only that newcomers became "fuller" participants. Because situated-learning theorists have viewed learning as socialization and have held that processes of socialization are universal, they have not attended to the possibility that the structures of communities of practice might vary. Thus, although situated-learning theory has highlighted action, it has not done so in a way that links environment to structure.

A third perspective, found in the research on helping (Perlow 1997, 2001; Perlow et al. 2004) and information seeking (Morrison 1993, Borgatti and Cross 2003, Cross and Sproull 2004), also views learning as a product of social interaction, but unlike situated learning theory, it foregrounds individuals turning to others for assistance. Rather than treat learning as role socialization, students of information seeking and helping have primarily been concerned with explicit knowledge, both social and technical (Morrision 1993). Consultations are often conceptualized as dyadic encounters, in which one individual (the target) explicitly seeks out another (the source). Much of the research on helping and information seeking has attempted to predict who consults whom by taking into account properties of relationships such as physical proximity, accessibility, or trust as well as aspects of organizational structure such as formal position. For instance, Cross and Sproull (2004) reported that in a consulting firm, employees looked to their hierarchical superiors for help in problem solving, rarely seeking counsel from their peers.

Because students of helping and information seeking were concerned with the networks that dyadic interactions inscribe, they explicitly attended to the social organization of learning. As a whole, these scholars have established that the structure of learning networks varies. What students of helping and information seeking have rarely done is attempt to link variation in the social organization of learning to differences in environments. An important exception can be found in the work of Perlow (Perlow 2001, Perlow and Weeks 2002, Perlow et al. 2004), who studied software development teams in India, China, the United States, and Hungary. Finding that networks inscribed by who helped whom varied significantly by country, Perlow et al. (2004) argued that the social organization of helping reflected important national differences in government policy, labor markets, and systems of education.

None of the foregoing perspectives concurrently considered environment, action, and the social structure of learning. The perspective rooted in March's work addressed environment and action (defined as decision making), while paying little heed to structure. Situated-learning theory highlighted action over environment and structure. The information-seeking and helping literatures built structure from action, leaving aside environment. In short, none of the three perspectives kept environment, action, and structure in equal play. Moreover, the three perspectives promoted different images of the number of learners and teachers in learning episodes and tended to discount purposeful and explicit teaching. March's learners appeared as socially influenced, but otherwise solitary, sense-makers. Situated-learning theorists stressed learning in the group context of a community of practitioners, with participation, not teaching, as the primary conduit of knowledge. Students of helping and information seeking conceived of learning almost exclusively as dyadic exchange and, in so doing, distinguished between sources and targets. In general, however, these scholars attributed initiative for learning solely to targets; consequently, they emphasized information seeking over information offering. We borrow from and build upon all three perspectives to construct a pragmatic conception of learning as a social activity, which allows us to study learning as a link between environment and the social organization of work.

Our pragmatic approach begins with a focus on learning explicit (rather than tacit) knowledge in the workplace. We focus on explicit knowledge because research on helping and information seeking has demonstrated that doing so enables one to identify the sources and targets of learning, a necessary first step for mapping the structure of learning networks. Ethnographic studies have also shown that individuals often gain explicit knowledge at work. Orr's (1996) ethnography of copier repair technicians is replete with accounts of how technicians, in consultation with documentation or, more often, with colleagues, learned explicit knowledge in the course of resolving technical problems. Similarly, Edmondson (1999) documented that groups in a manufacturing plant experimented, talked about errors, and asked other groups for help to gain explicit knowledge to solve a problem.

Our pragmatic notion of learning allows intentionality on the part of those who offer as well as those who seek knowledge, thereby giving equal opportunity to teaching. Recognizing that individuals purposefully teach at work enables one to ask whether teachers teach in different ways. Orr (1996) demonstrated that technicians sometimes taught each other by telling war stories of repair. At other times they taught by going to the site of a malfunctioning machine and guiding their colleague through an analysis. As Cross and Sproull (2004) discovered, sources sometimes provided targets with direct answers to queries. On other occasions they pushed the target to reexamine the problem and to consider broader consequences. Some forms of teaching require delving into details and, therefore, might consume more of the teacher's time than providing a quick answer to a clear question. In short, environmental differences may affect how teachers teach, what they teach, and how long it takes them to teach.

In sum, to explore how an organization's environment might configure its structure by shaping how individuals learn, we adopt a pragmatic conception of learning as a social activity that highlights the learning of explicit knowledge; acknowledges targets and sources; allows for intentionality on the part of either target or source; takes into account what is taught, how it is taught, and how much time was spent teaching it; and pays attention to the social configuration of learning. Identifying these attributes of learning demands an empirical technique for separating episodes of teaching and learning from an ongoing stream of behavior. The work of Roger Barker, a social psychologist who was interested in learning in schools and everyday contexts, proved useful for crafting such tools.

Barker's Techniques for Studying a Stream of Behavior

In the late 1940s and 1950s, Barker devised a painstaking approach for studying what he called the "stream of behavior" (Barker 1963). His method entailed following a subject, usually a child, throughout the course of a day, while meticulously recording all of the person's behaviors as well as where the action occurred, who was present, who said what, and what objects supported the person's activities. Although Barker and his students were influenced by anthropology (see Schoggen 1989), and their methods resembled participant observation, their approach to analyzing data differed significantly from how most ethnographers treat field notes. Whereas ethnographers look for meaning and themes and seek to portray the insider's perspective, Barker and his associates parsed their "specimen records" (i.e., fieldnotes) into "behavioral episodes," distinct units of analysis whose frequency could be counted and whose distributions could be analyzed. Using these counts and distributions, Barker would compare patterns of activity in different settings. In anthropological terms, Barker offered an etic rather than an emic analysis of action. The value of Barker's approach is that it allows one to document actions precisely over time and space, which is difficult to achieve when analysis targets meanings and themes. Accordingly, Barker's approach enables researchers to describe and systematically compare larger patterns of behavior that emerge from discrete actions and interactions in different contexts.

Barker defined a behavioral episode as a coherent run of behavior in which the constituent acts have a "constant direction," a purpose (Barker and Wright 1955, p. 5). Examples included "writing spelling lesson at blackboard" and "waiting for teacher to check spelling" (Barker 1963, pp. 291–299). Barker and his colleagues analyzed behavioral episodes by their internal structure, including what triggered the episode, how long it lasted, and how it ended. Analyzing behavioral episodes in this way allowed Barker and his associates to inventory types of actions associated with social scenes, which Barker called "behavioral settings." Key behavioral settings included churches, schools, drugstores, and homes. Barker and his colleagues showed that the mix and frequency of behavioral episodes associated with a kind of setting varied across locales. For example, Schoggen et al. (1963) reported that children in an American elementary school produced 1.33 behavioral episodes per minute, whereas children in a comparable English school produced 0.86 per minute. In Barker's terms, American and English elementary schools were similar behavioral settings with very different "behavioral ecologies." For Barker, a behavioral ecology was defined by a pattern of episodes and a distribution of actors across those episodes.

Barker's approach to analyzing behavior offers a way to separate episodes of learning and teaching explicit knowledge from the surrounding stream of action. Parsing learning episodes from the stream of behavior requires recognizing cues that mark the beginnings and endings of learning episodes. John Dewey's (1938) observation that learning often occurs when problems stymie progress helps on this score, for it suggests attending to cues that signal a breach in the action of a task, accompanied by a subsequent search for solutions to some perceived problem. Previous studies of work indicate that such cues are common and observable. Hutchins (1991) noted that cracking voices and perspiration were early signs of confronting an engineering breakdown on a large ship. Barley (1988) was able to identify the beginning of problem-solving episodes by noting unexpected expletives and machines that stopped working. Among knowledge workers, cues of trouble might include hands that stop typing, pencils that come to a standstill, or fingers that drum desktops. The endings of learning episodes are also marked by behavioral cues. For example, a person might close the book she was reading or take leave of the person she was consulting.

Having separated learning episodes from the stream of behavior, employing Barker's approach to analyzing episodes is relatively straightforward. One first classifies the episodes by their internal structure, including what was learned, who initiated the episode, who taught, and who learned. One then uses these features to count or catalog episodes to produce a profile. By a *teaching-learning ecology* we shall mean such a profile, which includes a pattern of learning episodes and a distribution of actors across those episodes. An important part of an ecology is the setting's teaching and learning network, which traces flows of knowledge from teachers to learners.

Scholars before us have used the term ecology in reference to learning. Levitt and March (1988, p. 331) used the term "ecologies of learning" to denote that when an organization learned it changed the context in which its competitors learned. Their notion is relational, but it differs from ours in that our learners are not competitors. Additionally, in Levitt and March's ecology, teaching is only incidental: One learner observes and imitates what another has done without the other intending to teach anything. Gherardi (2006, pp. xii, 192) also employs the term "ecology of learning." For Gherardi, knowledge is distributed across humans and nonhuman material objects. The question of interest is how knowledge is transferred and transformed via interactions in a network of humans and objects. Although we include objects in our analysis, we pay greater attention to the roles that people play.

We employ this approach to compare the teaching-learning ecologies of two engineering occupations: structural engineering and hardware engineering. The environments of these occupations differ in two striking ways that we expected would constrain and afford learning activities. First, knowledge changes much more slowly in structural engineering than in hardware engineering. Structural engineers draw on an understanding of physical principles that their field has been accumulating for more than 200 years. Hardware engineering is, by contrast, a young discipline whose knowledge changes quickly in the face of continuous, and often dramatic, technological advances. Second, state licensing requirements mandate that junior structural engineers practice under the guidance of a senior engineer for six years; no such requirement governs hardware engineers, who are largely unlicensed. We suspected that these two environmental differences portended a greater teaching role for senior structural than for senior hardware engineers and that learning networks in the two occupations would be differently structured.

Methods

Research Design

Engineering is an especially fruitful domain for studying how individuals gain explicit knowledge at work because technical learning, formal knowledge, and problem solving are integral to engineering practice (Vincenti 1990, Vaughan 1996, Henderson 1999). From among engineering's many disciplines, we chose to study structural engineers who design buildings and hardware engineers who design digital chips.

Structural engineers specify the materials, shapes, and sizes of the beams, piers, and other elements that transfer building loads to the ground to prevent buildings from collapsing. Hired by architects for specific projects, structural engineers create alternative designs. After the architect and building owner choose a design, structural engineers analyze the design and produce detailed drawings and calculations for contractors. Design and analysis also involves exchanging technical information with architects and contractors. Structural engineers must present their final design to government review boards that issue building permits.

Hardware engineers craft the logic of microprocessor cores, buses, and other chip components. Working on project teams for each version of the processor, hardware engineers create representations of these components by writing code in high-level programming languages that specify how each component will handle instructions and interact with other components. They also write programs to test and verify that their components function correctly.

To ensure that our findings were not particular to a given firm, we studied three structural engineering and three hardware engineering firms in the San Francisco Bay area. The structural engineering firms were homogenous professional firms in the sense that the only profession represented was structural engineering. Two of the firms had a staff of 15–20 structural engineers; the other had only four engineers at the site we studied (and eight others spread across two other sites). The remaining staff at each firm was small and consisted primarily of an office/payroll manager and a receptionist. One firm specialized in seismic upgrades. The second worked with developers on multistory commercial buildings and the third specialized in single-story fabrication plants for the computer chip industry.

All of the hardware engineering firms engaged in the marketing, design, and sale of products. One of the firms created microprocessor architectures and cores, another developed customizable microprocessor cores and peripherals, whereas the third designed programmable logic devices. All employed large numbers of software and hardware engineers in their design groups, but also had sizable sales and marketing staffs, as well as individuals employed in finance, accounting, human resources, office management, and reception. Firm sizes ranged from 200 to 2,000 employees, with the number of hardware engineers ranging from 30 to more than 100.

Across the six firms, we shadowed 19 structural engineers and 21 hardware engineers on repeated occasions. Most informants volunteered to be observed following presentations in which we explained our study; others volunteered after the study commenced. We sought to observe between 5 and 10 engineers at each site to ensure we captured a range of work and work behaviors. To maintain a focus on engineering rather than management tasks, we targeted junior engineers with fewer than 7 years of work experience and midlevel engineers with 7–15 years of experience. Specifically, we shadowed 1 senior (greater than 15 years experience), 5 midlevel, and 13 junior structural engineers; and 2 senior, 11 midlevel, and 8 junior hardware engineers. Between 1999 and 2002 we conducted 84 observations of structural engineers and 86 observations of hardware engineers. Although we shadowed senior engineers less than other engineers, we had many opportunities to observe them because they frequently interacted with junior and midlevel engineers. Senior engineers constituted only 8% of the people we shadowed, but they represented 33% of the engineers who appeared in the learning events we documented.

Data Collection

Engineering work has been notoriously difficult for sociologists to study (Downey et al. 1989, Barley 2005). Engineers speak in technical tongues specific to their specialty, they render mathematical calculations quickly, and they use sophisticated tools whose functioning is not easily gleaned. To make matters worse, understanding why engineers view a given situation as problematic requires scientific and technical knowledge. As a result, observers must work hard to hear what is said and to notice what is done. We took steps to meet these challenges. Before entering the field, our research team read texts from relevant civil and electrical engineering classes. Professional engineers (not in our study) tutored us on tasks and tools that we would encounter. Initially, two researchers jointly shadowed the engineers: One recorded events, whereas the other documented only technical terms. From the latter's field notes we developed technical glossaries to help us better hear and understand what was later said in solo observations.

We also had to develop special techniques for recording an engineer's stream of behavior. We found that attending to all the details of an engineer's work made it difficult for us to observe longer than three or four hours a day. In addition to recording time and taking running notes on the engineer's interactions with tools, people, and documents, we audiotaped conversations when we could not keep pace by writing notes or when discussions became very technical. When our informants worked at computers, we requested screen shots of software interfaces and digital copies of the documents with which they worked. We photocopied physical documents that the engineers had employed such as sketches, pages from brochures, and handwritten calculations.

Collecting so many types of data enabled us to prepare field notes that described actions, conversations, and visual images simultaneously, and, thus, to produce a record not only of what engineers did and said, but also of what they worked on. The first step in assembling a day's field notes was to expand the running notes taken in the field into full narratives. We transcribed tapes directly into the body of the narrative at the point where the talk occurred. Similarly, we indexed screenshots and photocopies of documents at the point in the field notes where they were used. Weaving together actions, conversations, and images allowed us to better understand an engineer's gestures, movements, and especially the referents of indexical speech (e.g., "Right here we need…"). Completing a full narrative of a day's observation took between two and two and a half days.

Over the course of the four years required to assemble the data on the six firms, we employed seven graduate students as fieldworkers. The intensity of our techniques for collecting and recording data prompted us to study the six firms sequentially so that the research could remain manageable. Three graduate students assisted with the fieldwork in structural engineering, and four others assisted with the research on hardware engineering. The first author did field work in all six sites and completed 50% of the observations in both occupations to ensure that we would retain a deep understanding of each site when the students who worked on the project graduated.

To ensure consistency in a project spanning so many years and sites, we mandated that all field notes conform to a standard format for noting action, indexing documents, and the like. The first author trained all fieldworkers and reviewed every set of notes that they produced for thoroughness, technical accuracy, and conformity to formats.

Data Analysis

The first task of data analysis was to separate learning episodes from the stream of behavior recorded in the field notes. Episodes proved relatively easy to identify because they were naturally bracketed within the flow of action by cues that signaled their beginnings and endings. Learning was always prompted in one of two ways, each of which interrupted the flow of an engineer's work. The first way was when the engineer encountered a problem in his own work; the second was when someone else offered to impart knowledge without having been asked to do so. Cues that engineers had encountered a snag in their work were hard to miss. Sometimes they swore, sighed, banged the table, or kicked the CPU. More frequently, they simply ceased typing or other activities in which they were involved and announced that they needed someone's help. At this point the engineers' work turned and proceeded in a different direction. They searched the Web, consulted a book, hunted down a colleague, or located a document from a previous project that they hoped would provide an exemplar. Learning episodes triggered by an individual who offered to share knowledge with an engineer were even more obvious. The person announced her intent by saying "Did you know ...," "I heard something interesting ...," and similar invitations to stop work and listen. Learning episodes of both types ended in a variety of ways. Some came to a close when the engineer solved or abandoned the problem and subsequently turned to another task. Others ended when the teacher departed. The remainder ended more subtly, perhaps coincident with the beginning of new episodes, such as when, in the course of a long design meeting, a different engineer took over the teaching or instruction changed from one topic to another.

Note that the cessation of a learning episode did not require that the engineer solve his or her problem. Nor did it require evidence that he or she had in fact learned something. Learning, as a behavior, is an activity, not a cognitive end state. What we counted as learning episodes were learning opportunities—times when new information or knowledge was presented to or sought out by the learner. We were not able to observe retention and reuse of the knowledge, the documentation of which would have required constant observation over weeks and perhaps months.

The following précis of a long learning episode drawn from our field notes in a hardware engineering firm illustrates how an episode appears in, and can be isolated from, a stream of action:

> Rick (*a midlevel hardware engineer*) was writing a diag (*short for "diagnostic program"*), but had a number of questions about how to do so, including when pdtlb (*the name of a variable*) took on the value of probetlb (*another variable*). Rick said that Randy (*a junior engineer*) could help him and went to search for him. He found Randy playing foosball in the break room and asked, "I need to build some probe diags to check the interlock feature. Think that will be too difficult?" Randy replied that it could probably be done and offered to help. The two men returned to Rick's cube, where they worked together for 25 minutes. Randy guided Rick through each line of code in the diag, answering questions as they went along. They tested the completed diag, but it failed. They debugged it and ran it successfully. Randy left and Rick continued with his work.

Having isolated episodes, we began identifying their relevant features by reading and comparing episodes from a subset of the notes. Ultimately, we identified seven attributes that were present in each episode and that jointly conveyed the episodes' social structure and technical content. Six attributes pertained to *how* learning unfolded: Whether the learning opportunity was sought by the learner or offered by a teacher, the social configuration of the learning episode, the rank of the learner, the rank of the teacher, the teacher's method of teaching, and the duration of the learning episode. The remaining attribute was the content of the episode or *what* was learned. Having identified the seven common attributes, we returned to the subset of episodes to determine the values that episodes could evince for each attribute. The values for the attributes comprised the codes that we subsequently assigned to all episodes in our field notes. Hence, every episode was described by the following seven values.

Opportunity Sought or Offered. When problems triggered learning, learners always initiated the search for knowledge. It would have been impossible for potential teachers to initiate such episodes because teachers rarely knew that the learner had encountered a problem until the learner alerted them. When individuals with a desire to teach triggered learning, the opportunity to learn was offered rather than sought. This distinction constitutes the first attribute of a learning episode, which we captured by noting whether learning was *sought* or *offered*.

Social Configuration. Learning episodes also varied by how many people were involved. Some involved an engineer learning alone, as when an engineer accessed a FAQ page on the website of a tool manufacturer. In other episodes, a single engineer interacted with just one other person, for example, when a colleague instructed him on using a tool. Still other episodes involved three or more people, as when an engineer reported in a staff meeting about a conference she attended. We coded these configurations *solo*, *dyadic*, and *group* episodes, respectively.

Learners' and Teachers' Ranks. We categorized learners and teachers in each episode by their rank: *junior* (0–6 years experience), *midlevel* (7–15 years experience), or *senior* (16 years or more experience) engineer. Other individuals also engaged in learning episodes with the engineers we observed. We categorized these individuals by their organizational membership. Thus, *internal other* referred to individuals such as CAD operators or sales staff within the firm, whereas *external other* referred to clients, vendors, and the like who were employed by outside organizations. In group episodes, learners and teachers could be of several ranks. For example, the learners (or teachers) in a group episode might consist of only juniors, a mix of junior and midlevel engineers, and so on. In those cases in which the learner consulted an artifact (such as a book), we coded the teacher's rank as an *artifact* so that we could track how often artifacts served as sources of knowledge.

In combination, the ranks of learners and teachers enabled us to capture how knowledge flowed among engineers within the firm. When combined with the sought and offered codes, the ranks of teacher and learner enabled us to determine how often each rank requested and offered knowledge. In particular, we could construct adjacency matrices based on the relationships "Rank X sought knowledge from rank Y" and "Rank Y offered knowledge to rank X." These matrices represent the social structure of each occupation's teaching-learning ecology.

Methods of Teaching. Across the body of episodes, teachers employed five distinct methods for teaching. One method was *tutoring*, in which the teacher adopted a pedagogical attitude. In practice, tutoring meant that teachers digressed from the particulars of the problem to extract larger lessons from the situation. When tutoring, rather than simply offering help, teachers typically quizzed the learner on fundamental concepts. In the following example, Cindy, a structural engineer, had called over her cubicle wall to ask Sam if he knew how she might resolve a discrepancy in her design between the size of the building's braces and the columns to which they were attached. Rather than answering her question directly, Sam began to quiz her about the struts that spanned the columns:

> Sam: [*from his cubicle*] Where's your gravity load on the end of the strut? [*Sam comes over to Cindy's cube and stands behind her, looking at her computer screen as she sits below him. A software package for analysis is open on the screen; it displays building diagrams with output data.*] Which way did you apply the seismic load? [*Cindy points beside a diagram to indicate the direction.*] Did you run gravity only?
>
> Cindy: No. [*She begins to navigate to a screen that will allow her to run an analysis.*]
>
> Sam: [*Showing irritation in his voice, he stops her.*] Let's look at the model first before you run the analysis. [*Sam leans over and points on the screen to a beam in the lower-left corner of a diagram and begins to quiz her.*] Is this in compression or tension?
>
> Cindy: [*hesitantly*] Tension?
>
> Sam: [*signaling a wrong answer and suggesting that she think again*] Under gravity loads?
>
> Cindy: Oh, compression, I'm sorry.
>
> Sam: So what's that one? [*Sam points to the upper-left horizontal beam in the diagram and continues the quiz.*] Compression or tension?
>
> Cindy: [*again hesitantly*] Tension?
>
> Sam: Yeah, now is this the same load case? [*Sam points to another set of beams on the opposite side of the diagram and offers a clue.*] The wind controls in this case.
>
> Cindy: [*explaining why the diagram can't be used for this quiz because the values on the screen combine gravity and lateral loads*] I'm showing the load combination.
>
> Sam: Show just the gravity. [*Cindy alters data in a table. As a result, the load values for some of the beams change on the screen. Sam points to the upper-right corner.*] So what's this? [*He directs her attention to the junction of the strut and the column and finally tells her that she could have passed his quiz by recognizing a fundamental feature of her structure.*] Are you visualizing the equilibrium of this point? This is the only horizontal component so it is in compression. Ok?
>
> Cindy: Yes.

When using a second method of teaching, *providing information*, teachers simply relayed facts, values, due dates, and similar information to the learner. Providing information usually happened quickly because the encounter required minimal instruction or guidance, as in the following example. Randy, a hardware engineer, and his office mate, Albert, sat in their shared office working on their computers. An engineer somewhere else in the company had reported a bug in Randy's code. Randy had just received an e-mail about the bug via the firm's bug-tracking software.

> Randy: [*Muttering as he reads the message*] Jim filed a bug on me! Every time he does, I can't understand what he's talking about.... [*Randy examines code that has been inserted into the e-mail. He asks Albert about a variable in the code that he does not recognize.*] Albert, SysIOPAddr, what is it?
>
> Albert: [*pauses on his laptop and turns his head toward Randy*] We have this concept of an I/O block now. The physical address is the start of the I/O block. All peripherals. [*Albert has just explained that SysIOPAddr, which stands for System Input-Output Physical Address, is a new variable that specifies the location of the I/O block.*]

When *instructing,* teachers offered learners brief instructions on how to do a task or solve a problem, rarely stopping their own work for more than a few minutes. Sometimes a teacher rattled off instructions without leaving her desk. In this example, George, a structural engineer, told Dennis how to prepare two foundation plans, one with caissons and the other with piles, both of which are structures for transferring building loads into the ground.

> George: We probably want to say, "Use these pile caps and throw in this amount of rebar as baseline," based on which one is cheaper, or meets all their scheduling problems. In general, that will determine which one they select. When one is selected, we'll begin the task of changing the drawings and doing the real pile caps, because three-feet-diameter concrete caissons is going to be a different spacing [*than the drawings currently show*]....
>
> Dennis: [*who has never prepared such a summary before*] So base the steel on...?
>
> George: [*correcting him*] Base the [*with emphasis*] *rebar*!
>
> Dennis: Yeah, the rebar.
>
> George: Base the rebar just on a wild-assed guess.
>
> Dennis: [*laughs*] Ok!
>
> George: [*more seriously*] Based on previous jobs that had pile caps. And that's just a....
>
> Dennis: [*interrupts and guesses the end of the sentence.*] Just a general note?
>
> George: It might be an 8 1/2 by 11 sketch that goes in with the bids. If we can describe it in words, that's even better. If it [*the rebar estimate*] gets sort of detailed, then

we probably need a sketch. To get accurate numbers on these things, you would need to go through a full design and detail them all out.

Collaborating occurred when the teacher worked with a learner on a problem. For this method to apply, teachers had to become absorbed in the problem, typically leaving their desks to accompany the learner to his workspace where they examined the problem in detail and often at length. In the excerpt below, Phil had come to Eric to ask about the order in which a piece of code handled microprocessor operations. Eric ultimately answered Phil's question, but not until they worked through the logic of the code together.

Phil: So the question is, at compile time, do you know the order that operations will come out on the processor interface for multiple list ordinates? [*By "you" Phil did not mean Eric, but rather Eric's code. Phil wants to know if Eric's code will be aware of the serial order in which the processor will start multiple operations.*]

Eric: Yes.

Phil: So the order is deterministic? [*Phil means that the order is not random.*]

{*Deleted passage in which Eric asks clarifying questions and Phil explains to Eric how he is writing his code. In the next passage, Eric's use of the word, "I," to refer to Phil's code is a signal that he has taken on the problem as his own.*}

Eric: Okay. Now, will this [*Phil's code*] check the cases, like, so I have a stream of loads and stores and I want to make sure that the data that I see is always the most up to date. Can I issue stores and loads all over the place on the processor interface?

Phil: Yeah.

{*Deleted passage in which Eric and Phil ask each other questions as they jointly analyze the problem until Phil has the answer to his question.*}

Phil: [*summarizing what he has learned from Eric*] If I do a load here and a load in a subsequent instruction, it will go out in that order.

Eric: The pipeline's blocking though on load, so you won't get to the second load until the response for the first one is done.

Phil: So why would pre-fetches reorder?

Eric: It's a stupid implementation.

Finally, when engineers were involved in interactions with people outside of their group (such as negotiating with a vendor), they sometimes modeled professional behavior for colleagues who had not yet mastered such situations. *Modeling professional behavior* occurred when teachers invited learners to participate in an important meeting so they could witness the interactions. After the event, teachers might call learners' attention to specific behaviors they deemed important. For example, in a break during a long phone negotiation with an architect about a beam, a senior structural engineer remarked to his junior colleagues, "What is going on here is that Frank [*the architect*] is asking SD [*an earlier stage of design*] questions, but we are in DD [*a later stage of design*]. His idea for cutting the TJI [*a kind of beam*] and putting four-by-fours sticking out is bad and expensive. He wants it for looks." Although implicit modeling occurred continually, only on these special occasions did engineers acknowledge them as learning opportunities.

Duration. We time-stamped our field notes roughly every 10 minutes. With a limited amount of interpolation, we were able to determine the duration of learning episodes in five-minute categories, with one special category for episodes that were less than a minute. We placed episodes in this "under a minute" category if they consisted of no more than two short lines of dialogue (e.g., a quick question with a short answer) or described a quick action. To facilitate calculation of average episode duration for each occupation, we assumed the midpoint for each category. For example, episodes in the category "1–5 minutes" were counted as having lasted 3.5 minutes. We also calculated average episode duration by teaching method to investigate whether some learning takes longer than others.

Content. Learning episodes evinced five broad types of content: technical concepts, tools, procedures, politics, and management. Learning episodes concerning *technical concepts* revolved around theories, data, technical heuristics, and other topics in engineering, mathematics, or science. Episodes about *tools* centered on technologies such as simulators and analysis packages. Episodes regarding *procedures* were about learning work rules and routines, such as the rules for uploading files to a common code repository. Learning episodes dealing with *politics* concerned the responsibilities of members of an engineer's role set, how these individuals viewed the world, and how to negotiate with them. When *management* was the content, episodes covered topics like budgeting and scheduling.

Using Atlas.ti™ software, the first author and two research assistants assigned the preceding codes to each learning episode. To ensure consistency, the three coders each initially coded a subset of the episodes from each occupation and discussed their differences until they came to agreement on the proper interpretation and application of each code. Each then coded a second subset of episodes. After the coders achieved substantial agreement on the second subset for each occupation, they split up the remaining field notes and proceeded to code the full set of 534 learning episodes: 307 episodes for structural engineering and 227 for hardware engineering.

Our discussion of results begins by studying action, which we accomplish by analyzing and inventorying the attributes of each ecology's learning episodes. We then show how action shapes structure by synthesizing differences in the two occupations' patterns of episodes to delineate two radically different teaching-learning ecologies. Finally, we explore environmental differences that seem to account for differences between the ecologies.

Teaching-Learning Ecologies

Studying Action: Analyzing Attributes of Learning Episodes

Social Configuration. Table 1 shows the distribution of social configurations of learning episodes in each occupation. Dyadic episodes were predominant in both, comprising 71% of all episodes in structural engineering and 74% in hardware engineering. Group learning was more common among structural engineers (24%) than among hardware engineers (9%), who were more likely to teach themselves (17% versus 5%). Hardware engineers expected colleagues to attempt to resolve problems on their own before approaching others. Rick, a midlevel engineer, explained how engineers would sanction those who asked for help too soon, "If there's a problem in the build [processor code] and you immediately fire off e-mail, you'll get flamed. A bunch of people used to do that…. And if they did that and there was an obvious problem that they should have been able to see, they'd get abused."

When an engineer decided to solve a problem on her own (a solo episode), she could explore possible solutions through trial and error or she could consult a manual, webpage, or other artifact for information. In the first case, it is reasonable to say the engineer taught herself; in the second, the artifact served as a source of knowledge. Table 2 indicates that in both occupations engineers usually turned to artifacts rather than puzzle through problems on their own, a result that was stronger in hardware engineering (87%) than in structural engineering (67%).

Seeking and Offering Knowledge Across Ranks. Whether a person seeks or offers knowledge has meaning only in dyadic and group episodes. Table 3 displays for each occupation the percentage of episodes in which learners sought and teachers offered knowledge in dyadic episodes, in group episodes, and overall. The differences are striking. Panel A of Table 3 shows that in dyadic episodes in structural engineering, learners sought knowledge equally as often as teachers offered it. By contrast, in dyadic episodes in hardware engineering, learners sought knowledge much more (80%) than teachers offered it (20%). Teachers in structural engineering took an even more active role in group episodes (Panel B), offering knowledge in 75% of all episodes. In hardware engineering, learners still sought more (56%) than teachers offered (44%), albeit the difference is smaller than in dyadic episodes. Overall, as shown in Panel C, in structural engineering teachers offered knowledge (56%) more often than learners sought it (44%), whereas in hardware engineering, nearly 80% of the episodes began with a learner seeking knowledge. These results suggest a much more learner-driven ecology in hardware engineering than in structural engineering.

An engineer seeking knowledge in a dyadic episode could approach a junior, midlevel, or senior engineer.

Table 1 Social Configuration of Learning Episodes by Occupation

	Structural engineering		Hardware engineering	
	Percentage (%)	*N*	Percentage (%)	*N*
Solo	5	15	17	39
Dyadic	71	219	74	168
Group	24	73	9	20
Total	100	307	100	227

Table 2 Teachers in Solo Episodes by Occupation

	Structural engineering		Hardware engineering	
	Identity of teacher		Identity of teacher	
Rank of learner	Self	Artifact	Self	Artifact
Junior	2	10	3	20
Midlevel	3	0	2	13
Senior	0	0	0	1
Total	5	10	5	34
Column (%)	33	67	13	87

Table 3 Knowledge Sought and Offered by Occupation

	Structural engineering		Hardware engineering	
	Percentage (%)	*N*	Percentage (%)	*N*
Panel A: Knowledge sought and offered in dyadic episodes by occupation				
Sought by the learner	50	110	80	134
Offered by the teacher	50	109	20	34
Total	100	219	100	168
Panel B: Knowledge sought and offered in group episodes by occupation				
Sought by the learner	25	17	56	10
Offered by the teacher	75	51	44	8
Total	100	68	100	18
Panel C: Total knowledge sought and offered by occupation				
Sought by the learner	44	127	77	144
Offered by the teacher	56	160	23	42
Total	100	287	100	186

Notes. Not included in Table 3 are episodes in which a dyad or group learned from a resource. The addition of five such episodes in structural engineering and two in hardware engineering bring the totals in Panel C to 292 and 188, respectively, which match the values in Table 1.

Alternatively, he could approach internal or external others. Teachers had the same options for offering knowledge. The first five rows and columns of each occupation's section in Panel A of Table 4 display the matrix of who sought knowledge from whom, organized by the rank of teacher and learner. The values in the boxed matrix give the number of learning episodes per hour of observation.[1] For example, in the structural engineering section of Panel A, junior engineers sought knowledge from other junior engineers at the rate of 0.27 episodes per hour. The sixth column and sixth row show the total number of learning episodes per hour by the rank of learner and teacher, respectively. In the case of structural engineers, juniors sought knowledge at the rate of 0.51 episodes per hour and provided knowledge at the rate of 0.29 episodes per hour. The seventh column and row, respectively, display the percentage of the total amount of learning and teaching displayed in the table by rank. Here, junior structural engineers learned in 55% of the 0.92 episodes per hour in which knowledge was sought, and they accounted for 32% of the teaching in these episodes. Panel B, which has the same structure as Panel A, shows to whom teachers offered knowledge. Panel C combines results for each occupation to show the total pattern of who learned from whom: For each occupation, the matrix in Panel C (the first five rows and the first five columns) is the sum of its matrix in Panel A (seeking) and the transpose of its matrix in Panel B (offering).

Panel A reveals that in both occupations junior engineers sought knowledge more frequently than did engineers of other ranks (55% and 52%). Among Panel A's episodes, seniors did the most teaching in structural engineering (54%), but comparatively little in hardware engineering (14%). In hardware engineering, juniors rather than seniors were the primary teachers in episodes where knowledge was sought (48%). In fact, junior and midlevel hardware engineers primarily sought knowledge from junior engineers. (The values in the first column are the highest in their respective rows for these two ranks.) By contrast, in structural engineering only juniors turned to juniors.

As Panel B shows, in structural engineering almost all offers to teach came from senior engineers (88%), who targeted junior engineers at about twice the rate that they targeted midlevel engineers (with rates of 0.42 and 0.23, respectively). In hardware engineering, midlevel engineers made the most offers to teach (42%) and taught engineers of all ranks. They were followed closely by senior engineers (40%), who primarily taught internal and external others. Junior hardware engineers were the least likely to offer to teach. As in structural engineering, junior hardware engineers received most offers (31%) followed by midlevel (22%) engineers.

Panel C highlights the sharp differences between the two occupations' teaching-learning ecologies. In structural engineering, seniors did most of the teaching (71%) and juniors predominantly learned (53%). In hardware engineering, juniors were not only the primary learners (48%), they were also the most likely to teach (41%). Moreover, the frequency of teaching in hardware engineering declined as rank increased (41% to 26% to 19%). In sum, Table 4 suggests that senior structural engineers had valuable knowledge, that they realized less-experienced engineers required this knowledge, and that they taught proactively. In contrast, junior hardware engineers had much to teach as well as learn; midlevel and senior hardware engineers also taught and learned, albeit less frequently.

Network diagrams highlight the stark structural differences between the two teaching-learning ecologies. Panel A of Table 5 dichotomizes the data in Panel C of Table 4 using a cutoff that eliminates learning rates of less than one episode per 45-hour week (on average, engineers in both occupations logged 9-hour days in the office). Multiplying the matrices in Panel A of Table 5 by those in Panel C of Table 4 after dropping internal and external others from both and then removing the diagonal (teaching and learning within a rank) yields the values from which we constructed the weighted directed graphs in Panel B of Table 5. The arrows in each graph indicate a flow of knowledge from teachers to learners.

The graph for structural engineering shows that knowledge flowed downward: Seniors taught junior and midlevel engineers, and midlevel engineers taught juniors. Seniors taught juniors six times more frequently than did midlevel engineers. Thus, the graph makes clear that senior engineers were predominantly responsible for teaching in structural engineering and that each rank played a unique role in the network of teaching and learning. In hardware engineering, hierarchy was absent and knowledge flowed among all ranks. Because all ranks taught and learned, each rank was structurally equivalent. In short, the graphs for structural engineering and hardware engineering, respectively, inscribe a system of role inequality versus a system of role equality.

Junior structural engineers were well aware of their role as learners under the guidance of senior engineers, whose advice they accepted with little objection. Senior structural engineers were similarly well aware of their role as teachers. For example, a senior structural engineer attested to the importance of seniors as sources of knowledge:

> The way we spread knowledge in this industry is from senior engineers to junior ones. A junior engineer might ask a senior engineer if he has a typical detail, or, more likely, he won't recognize that he even needs one. The senior engineer will look at his drawing and say, "You need a detail here. We have a typical detail for this." And then the junior engineer will just put it in, and maybe if he is good, he will ask the senior engineer why it is designed that way, or maybe the senior engineer will take the initiative to explain it to him.

Table 4 Seeking and Offering Knowledge in Dyadic Episodes by Rank and Occupation

Structural engineering							
	Teacher						
Learner	Junior	Midlevel	Senior	Internal other	External other	Total number of learning episodes per hour	% of all learning done per hour
Panel A: Learner seeks knowledge from teacher							
Junior	0.27	0.07	0.14	0.01	0.01	0.51	55%
Midlevel	0.00	0.00	0.18	0.00	0.00	0.18	20%
Senior	0.01	0.00	0.17	0.00	0.00	0.17	19%
Internal other	0.01	0.00	0.00			0.01	2%
External other	0.00	0.04	0.00			0.04	5%
Total number of learning episodes per hour	0.29	0.12	0.49	0.01	0.01	0.92 (sum)	
% of all teaching done per hour	32%	13%	54%	2%	1%		100%
Panel B: Teacher offers knowledge to learner							
Junior	0.05	0.01	0.01	0.00	0.01	0.08	8%
Midlevel	0.01	0.00	0.00	0.00	0.01	0.03	3%
Senior	0.42	0.23	0.17	0.00	0.05	0.87	88%
Internal other	0.00	0.00	0.00			0.00	0%
External other	0.01	0.00	0.00			0.01	1%
Total number of learning episodes per hour	0.50	0.24	0.17	0.00	0.08	0.99 (sum)	
% of all learning done per hour	50%	24%	18%	0%	8%		100%
Panel C: Learner-teacher network							
Junior	0.32	0.09	0.57	0.01	0.02	1.01	53%
Midlevel	0.01	0.00	0.41	0.00	0.00	0.42	22%
Senior	0.01	0.00	0.33	0.00	0.00	0.35	18%
Internal other	0.01	0.00	0.00			0.01	1%
External other	0.01	0.05	0.05			0.12	6%
Total number of learning episodes per hour	0.37	0.14	1.36	0.01	0.02	1.91 (sum)	
% of all teaching done per hour	19%	7%	71%	1%	1%		100%

Similarly, junior hardware engineers realized that they needed to teach. A junior hardware engineer had this role in mind as she explained how she assessed a learner's level of understanding, "Usually if people ask me questions, I'll say, 'Oh, did you read the spec?' I'm not trying to challenge them. I want to know: Do I start from ground zero or do you have a very specific question, and I should just give you an answer?" Because junior hardware engineers were sought as teachers, they had to judge when to document their knowledge to avoid having to repeatedly provide the same information. As a junior hardware engineer put it, "If someone comes and asks me something, I will tell him what the answer is. The second person comes, I will tell him again. The third person comes, and I ask, 'Did you read the spec?' And when the fourth one comes, I start writing a small summary, some documentation."

Although we could have also used rates of learning in group episodes, correcting for the number of groups we observed with each combination of members made this

Table 4 (cont'd.)

Hardware engineering							
	Teacher						
Learner	Junior	Midlevel	Senior	Internal other	External other	Total number of learning episodes per hour	% of all learning done per hour
	Panel A: Learner seeks knowledge from teacher						
Junior	0.31	0.09	0.07	0.01	0.04	0.52	52%
Midlevel	0.14	0.03	0.03	0.03	0.01	0.23	23%
Senior	0.01	0.04	0.00	0.04	0.04	0.13	13%
Internal other	0.02	0.04	0.04			0.10	10%
External other	0.00	0.01	0.00			0.01	1%
Total number of learning episodes per hour	0.48	0.21	0.14	0.08	0.09	0.99 (sum)	
% of all teaching done per hour	48%	21%	14%	8%	9%		100%
	Panel B: Teacher offers knowledge to learner						
Junior	0.02	0.01	0.01	0.00	0.00	0.04	15%
Midlevel	0.05	0.03	0.02	0.02	0.00	0.11	42%
Senior	0.01	0.02	0.00	0.04	0.04	0.11	40%
Internal other	0.00	0.01	0.00			0.01	3%
External other	0.00	0.00	0.00			0.00	0%
Total number of learning episodes per hour	0.08	0.06	0.03	0.06	0.04	0.27 (sum)	
% of all learning done per hour	31%	22%	10%	21%	15%		100%
	Panel C: Learner-teacher network						
Junior	0.33	0.14	0.08	0.01	0.04	0.60	48%
Midlevel	0.15	0.06	0.04	0.03	0.01	0.29	23%
Senior	0.02	0.06	0.00	0.04	0.04	0.16	13%
Internal other	0.02	0.06	0.08			0.16	13%
External other	0.00	0.01	0.04			0.05	4%
Total number of learning episodes per hour	0.52	0.32	0.25	0.08	0.09	1.26 (sum)	
% of all teaching done per hour	41%	26%	19%	7%	7%		100%

Note. Unless otherwise indicated as percentages, values are rates reflecting the number of learning episodes per hour of observation.

task numerically daunting. Because there were so few group episodes, we simply present the raw number of episodes of various types, which tells the story clearly enough. Table 6 shows group episodes for structural engineering: Panel A displays episodes in which learners sought knowledge, and Panel B presents episodes in which teachers offered it. Blackened cells reflect combinations that we have already discussed (dyadic episodes or episodes in which a single engineer turned to an artifact) or that we deemed unrealistic (engineers did not teach artifacts nor did artifacts offer to teach). Empty cells indicate zero episodes.

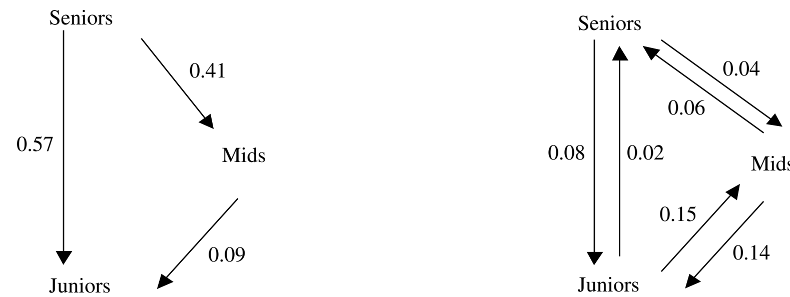
All but one of the nonzero cells in Panel A occur in the right side of the matrix, reflecting episodes in which seniors, internal others, external others, or artifacts served as the source of knowledge for groups that

Table 5 Learner-Teacher Network in Dyadic Episodes by Occupation

Panel A: Dichotomized learner-teacher network

Structural engineering				Hardware engineering			
	Teacher				Teacher		
Learner	Junior	Midlevel	Senior	Learner	Junior	Midlevel	Senior
Junior	0	1	1	Junior	0	1	1
Midlevel	0	0	1	Midlevel	1	0	1
Senior	0	0	0	Senior	1	1	0

Panel B: Dyadic learner-teacher network



Note. Values in the diagrams are rates reflecting the number of learning episodes per hour of observation.

included a junior engineer in every episode but one. The one outlying nonzero cell reflects four episodes in which a junior engineer taught a group of junior, midlevel, and senior engineers. Three of the four episodes involved providing information: Two of the episodes reflected brief reports in staff meetings on seminars attended by the junior engineers and one concerned the directory structure on the server. In the fourth episode, a junior engineer briefly instructed the group how to design a discontinuous bearing wall. The junior engineers, in short, never taught the group by collaborating, tutoring, or modeling professional behavior. By contrast, in the eight episodes in which seniors taught a group, they helped by collaborating in half of them and tutored in a quarter; for the remaining two, they modeled professional behavior in one and provided information in the other. Panel B shows that almost all episodes of offering entailed a single senior engineer teaching groups composed only of juniors (43 episodes). Again, when seniors taught the group, they primarily collaborated (25 episodes) and tutored (11 episodes). The single outlier reflects a junior engineer providing information on how to carry out an operation in a software package. Combined, Panels A and B reflect considerable teaching by senior engineers and considerable learning by juniors, mirroring the dyadic results for structural engineering. Table 7 displays similar results for group episodes in hardware engineering. As in structural engineering, seniors taught more than other ranks in group episodes, but learning was not concentrated among juniors. Teachers of all ranks primarily provided information, although on one occasion a senior engineer tutored the group and on another a midlevel engineer did.

Teaching Method. More generally, for dyadic and group episodes combined, Table 8 displays for each occupation the relative frequency at which teachers employed the five teaching methods. In neither occupation did teachers often model professional behavior. Structural engineers distributed their teaching more evenly across the remaining four methods than did hardware engineers. In structural engineering, collaborating was the most common teaching method (39%), followed by providing information (24%), instructing (18%), and tutoring (17%). However, junior structural engineers provided information over half the time they taught (as did internal and external others), whereas midlevel engineers predominantly collaborated. Collaboration was similarly the main method of senior engineers, for whom tutoring, which was rare among junior and midlevel engineers, was the next most employed method. In hardware engineering, providing information was the primary teaching method (74%), trailed considerably by collaborating (15%) and instructing (10%). Unlike structural engineers, hardware engineers almost never tutored (1%). Moreover, differences in teaching method by rank were small among hardware engineers.

The teaching methods used by midlevel and senior structural engineers reflected the need for side-by-side instruction. A senior engineer argued that seniors needed to teach juniors on the job because universities emphasized computer analysis rather than practical knowledge critical for doing structural engineering's work:

> There is a whole layer of analysis and computer activity that dominates time in school to the detriment of real-world, hands-on lab work in wood, steel, and concrete. When I was a student, we designed, built, and

Table 6 Seeking and Offering Knowledge in Group Episodes by Rank in Structural Engineering

Panel A: Learner seeks knowledge from teacher

	Teacher														
Learner	J single	J group	J & M group	J & S group	J, M & S group	M single	M group	M & S group	M, S & E group	S single	S group	I	E	A	Total
J single											1				1
J group										5					5
J & M group														1	1
J & S group										1			1	1	3
J, M & S group	4									1		4		2	11
M single															0
M group															0
M & S group															0
M, S & E group														1	1
S single															0
S group															0
I															0
E															0
A															0
Total	4	0	0	0	0	0	0	0	0	7	1	4	1	5	22

Panel B: Teacher offers knowledge to learner

	Learner														
Teacher	J single	J group	J & M group	J & S group	J, M & S group	M single	M group	M & S group	M, S & E group	S single	S group	I	E	A	Total
J single					1										1
J group															0
J & M group															0
J & S group															0
J, M & S group															0
M single															0
M group															0
M & S group															0
M, S & E group															0
S single		43		3	2										48
S group	2														2
I															0
E															0
A															0
Total	2	43	0	3	3	0	0	0	0	0	0	0	0	0	51

Notes. J = Junior, M = Midlevel, S = Senior, I = Internal other, E = External other, A = Artifact. Values in the cells reflect numbers of episodes.

tested ... [today] it is easier to lose the feel of physical aspects .... They don't know physical processes like mixing cement .... It takes longer to get people up to speed. The training is tedious, one-on-one with 28 engineers. Analysis should be 10% of the job, which means the typical student is only prepared for 10% of the job upon graduation.

In contrast, junior hardware engineers graduated from school knowing most of what they needed to know for their jobs because their skills and the occupation's division of labor were so highly specialized. Specialization meant that the design and analysis that newly minted hardware engineers would do at work was largely equivalent to what they had practiced in school. The problem was that each specialist's work was so tightly integrated with the work of specialists working in other areas that they often needed to understand aspects of the specific components on which their colleagues were working to do their own work effectively. Instead of learning related specialties, hardware engineers relied on each other to be specialists in the details, to provide crucial information at that point in time when the information was needed. One hardware engineer justified the approach, "Our plate's full and we don't really have enough time to learn about everything." Another illustrated the nature of the specialists' dependence on each other:

> Eric: Phil has this complicated math library for random number generation. So, if I have this code and I want to test it randomly, and I want to know how much randomness I need, then Phil will point at his little graphic and say, "You need to run 10 to 20 cycles."
>
> Observer: So how extensive does your own knowledge of randomness have to be?
>
> Eric: We rely on Phil, basically.

Table 9 presents the results of our analysis of how long learning episodes lasted by teaching method and

Table 7 Seeking and Offering Knowledge in Group Episodes by Rank in Chip Design

Panel A: Learner seeks knowledge from teacher

Learner	Teacher: J single	J group	J & M group	J&S group	J, M & S group	M single	M group	M & S group	S single	S group	I	E	A	Total
J single														0
J group													2	2
J & M group														0
J & S group	1										1			2
J, M & S group						1			3					4
M single														0
M group														0
M & S group	2								1					3
S single							1							1
S group														0
I														0
E														0
A														0
Total	3	0	0	0	0	1	1	0	4	0	1	0	2	12

Panel B: Teacher offers knowledge to learner

Teacher	Learner: J single	J group	J & M group	J & S group	J, M & S group	M single	M group	M & S group	S single	S group	I	E	A	Total
J single														0
J group														0
J & M group														0
J & S group														0
J, M & S group														0
M single			1					1						2
M group														0
M & S group														0
S single					5			1						6
S group														0
I														0
E														0
A														0
Total	0	0	1	0	5	0	0	2	0	0	0	0	0	8

Notes. J = Junior, M = Midlevel, S = Senior, I = Internal other, E = External other, A = Artifact. Values in the cells reflect numbers of episodes.

occupation. Providing information was the quickest method in both occupations, averaging 2.8 and 6.7 minutes per episode in structural and hardware engineering, respectively. In both occupations, instructing took twice as long, on average, as providing information; collaborating took three times as long. All average durations were greater in hardware than in structural engineering. The predominance of institutional knowledge in structural engineering and its relative absence in hardware engineering may have contributed to this difference. To provide information, for example, in structural engineering often merely required a statement of fact that may have been known to many (e.g., the load a beam of a certain weight and dimension could bear), whereas even simple facts in hardware engineering routinely required the recounting of a local and not widely known history (e.g., an explanation of what function a certain variable performed may have required explaining the problem it was meant to solve and so forth).

Content. Table 10 conveys the content of what was taught and learned for each occupation by rank. Over 50% of all learning episodes in both occupations concerned technical concepts. Conversely, management issues were rarely the focus of learning in either field. After technical concepts, procedures (23%) were the most common content in structural engineering, followed by politics (14%) and tools (8%). All structural engineers were most likely to teach technical concepts followed by procedures, but juniors' third-most frequent topic was tools, whereas politics held third spot for midlevel and senior engineers. In hardware engineering, almost all other teaching episodes were about tools (30%), a pattern that held across rank.

Table 8 Teaching Method in Dyadic and Groups Episodes by Rank of Teacher and Occupation

	Junior		Midlevel		Senior		Internal other		External other		Total for teaching method	
	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*
						Structural engineering						
Providing information	51	30	20	5	15	28	83	5	50	2	24	70
Collaborating	20	12	64	16	42	82	0	0	25	1	39	111
Instructing	22	13	8	2	19	36	17	1	0	0	18	52
Tutoring	7	4	4	1	22	43	0	0	25	1	17	49
Modeling professional behavior	0	0	4	1	2	4	0	0	0	0	2	5
Total for rank	100	59	100	25	100	193	100	6	100	4	100	287
						Hardware engineering						
Providing information	78	50	59	35	75	21	90	26	100	6	74	138
Collaborating	16	10	25	15	11	3	0	0	0	0	15	28
Instructing	6	4	14	8	11	3	10	3	0	0	10	18
Tutoring	0	0	2	1	4	1	0	0	0	0	1	2
Modeling professional behavior	0	0	0	0	0	0	0	0	0	0	0	0
Total for rank	100	64	100	59	100	28	100	29	100	6	100	186

Notes. Not shown in Table 8 are episodes in which the dyad or group learned from a resource. The addition of five such episodes in structural engineering and two in hardware engineering bring the totals to 292 and 188, respectively, which matches the values in Table 1.

Table 9 Duration of Episodes by Teaching Method and Occupation

	Structural engineering		Hardware engineering	
	Average duration (in minutes)	As multiple of providing information	Average duration (in minutes)	As multiple of providing information
Providing information	2.8	1	6.7	1
Collaborating	7.4	3	18.0	3
Instructing	5.3	2	13.8	2
Tutoring	6.0	2	43.5	6
Modeling professional behavior	40.5	14	0.0	0

Table 10 Content by Rank of Teacher and Occupation

	Junior		Midlevel		Senior		Internal other		External other		Total for content	
	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*	Percentage (%)	*N*
						Structural engineering						
Technical concepts	53	31	64	16	54	105	0	0	50	2	54	154
Tools	19	11	4	1	3	6	83	5	0	0	8	23
Procedures	25	15	20	5	22	42	17	1	50	2	23	65
Politics	2	1	12	3	18	35	0	0	0	0	14	39
Management	2	1	0	0	3	5	0	0	0	0	2	6
Total for rank	100	59	100	25	100	193	100	6	100	4	100	287
						Hardware engineering						
Technical concepts	64	41	63	37	75	21	59	17	0	0	62	116
Tools	31	20	29	17	14	4	31	9	100	6	30	56
Procedures	3	2	5	3	4	1	7	2	0	0	4	8
Politics	2	1	3	2	7	2	0	0	0	0	3	5
Management	0	0	0	0	0	0	3	1	0	0	1	1
Total for rank	100	64	100	59	100	28	100	29	100	6	100	186

Notes. Not shown in Table 10 are episodes in which the dyad or group learned from a resource. The addition of five such episodes in structural engineering and two in hardware engineering bring the totals to 292 and 188, respectively, which matches the values in Table 1.

That hardware engineers were so focused on learning about tools directly reflects the occupation's high rate of technological change. A hardware engineer commented on how engineers in his firm kept abreast of tool changes:

> One of the things about this work is that there are lots of tools. And most of these tools keep changing, like the simulator that we were using several years ago is not the simulator we're using today.... So we constantly have to keep updated. The tool manufacturers go to conferences where they present new tools. Every year people from our company go there and scout out new tools.

In structural engineering nearly every tool that practitioners used, including software programs, rested on fundamental physical principles that rarely changed. As a result, old tools were usually as useful as new tools, which reduced the need for learning about tools. Consider the building code, which was republished every few years with much of its content unchanged. As a structural engineer noted, not only did the code change slowly, but the code was often unnecessary for an engineer who understood the principles on which it was based:

> This is the book with the stupid codes. It is published every three years or so, and becomes effective three years after that, so this is the 1994 code, which became effective in 1997, and this one, the 1997 code, became effective in 1999. They make us follow the code instead of saying, "You guys have been in school for five years: Do your physics and leave us alone." It's sad.

Building Structure from Action: Two Very Different Teaching-Learning Ecologies

The foregoing differences in the attributes of learning episodes define the contours of two substantially different teaching-learning ecologies. Structural engineering's ecology was pedagogical. A significant proportion of the discipline's knowledge consisted of general physical principles, fundamental properties of materials, the functioning of basic components (steel beams, concrete piles, and so forth) and local building codes, all of which constituted well-codified tenets or facts. Gaining this knowledge accounted for the majority of learning. With each passing year, structural engineers added to their personal stock of knowledge to become more skilled at their craft. Senior engineers viewed new hires, and new hires viewed themselves, as incompletely educated. Understanding that mastering the craft required strong guidance, as mandated by law, junior, and even midlevel structural engineers rarely challenged seniors' opinions. Overall, teaching and learning in structural engineering resembled a system of masters and apprentices.

In this apprenticeship system, seniors were proactive. They orchestrated much of the learning by judging what junior and midlevel engineers needed to know and when they needed to know it. The seniors' power to control and script learning was illustrated by Sheila, a junior engineer, who had been waiting more than a year to learn a software package for analyzing concrete structures. Sheila estimated that it would take only four hours to learn the package, which she believed would increase her skills and broaden the set of projects to which she could be assigned. However, her supervisor refused to allow her the time to learn the tool because it could not be billed to her current project. Nor could Sheila learn the package by coming in after hours because the firm didn't want engineers working overtime. In this manner the senior engineer's plans for Sheila's learning took precedence over her own desires.

The master-apprentice ecology created a role structure of inequality marked by status differences associated with rank. In fact, as the graph of teaching and learning in Table 5 shows, the right to teach traced a strong hierarchical structure. Senior engineers directed the activities of junior and midlevel engineers, who, in turn, sought out seniors for guidance and deferred to their opinions. Midlevel engineers taught juniors, albeit far less frequently than did seniors. Junior and midlevel engineers almost never deigned to teach seniors. Differences by rank also manifested themselves in teaching methods: Tutoring was a method nearly unique to senior engineers, whereas junior engineers provided information fully half the time they taught and midlevel engineers overwhelmingly collaborated.

In contrast, the teaching-learning ecology of hardware engineering revolved around information sharing rather than pedagogy. Teaching in this context rarely involved tutorials on applying engineering fundamentals to specific instances of design. Instead, teaching and learning in engineering almost always entailed one engineer sharing specialized and often factual knowledge with another, regardless of rank. Engineers of all ranks, including juniors, held some type of specialized knowledge. No one expected hardware engineers to master all of this knowledge, but everyone expected them to know whom to consult when they needed information. As one hardware engineer put it, "Whenever I go to a new company, I quickly try to find out who is the Perl person, who is the Makefile person, and so on. So knowledge is knowing where the experts are."

Furthermore, unlike structural engineers, hardware engineers, who sought knowledge far more often than they were offered it, were largely in charge of their own learning. They determined what they needed to learn and when to learn it. Jason, a junior engineer, exhibited such independence when Phil, a midlevel engineer, tried to persuade Jason to join his project team. Jason, who had the right to turn Phil down, made clear that he would not join the team unless his tasks would afford him new expertise. This example shows a junior hardware engineer doing what a junior structural engineer never could: shop for a project.

Hardware engineering's teaching-learning ecology was populated with independent but interacting specialists (rather than masters and apprentices) who provided one another with information and help as needed. Specialists differed significantly from masters in that they rarely offered help without first being asked. Moreover, they did not try to choose what their peers learned. Instead, each specialist was responsible for maintaining her particular knowledge and acquiring whatever additional knowledge she required to do her job. This arrangement of distributed knowledge and independent learners supported a structure of equality in which rank had little relevance. As hardware engineering's graph of teaching and learning in Table 5 makes clear, all ranks were structurally equivalent.

Exploring Environmental Factors That Shape Action and Structure

Although we began by studying engineers in six firms, we found neither one universal nor six distinct models for learning. Instead, engineers in each structural engineering firm taught and learned their profession's explicit knowledge in similar ways. The same was true of hardware engineers: The patterns of teaching and learning of explicit knowledge in hardware engineering were the same across the three firms.[2] Thus, the data pointed to two occupational ecologies: one inhabited by structural engineers, the other by hardware engineers. Our informants and our observations of their work suggested that at least four environmental factors and one technological factor constrained learning activities across the two occupations, ultimately contributing to two different teaching-learning ecologies. The four environmental factors were the rate of knowledge change in the occupation, the nature of the market that the occupation served, licensing requirements, and the nature of the competition within the occupation's industry. The technological factor was the ability to judge the soundness of designs.

Rate of Knowledge Change. We chose to study structural and hardware engineering because we knew that knowledge changed slowly in the former and quickly in the latter. The rapid pace at which knowledge changed in hardware engineering strongly constrained learning activities. When knowledge changes rapidly, older knowledge becomes obsolete, undercutting the value of experience. As a result, seniority confers no benefit with respect to technical acumen. Because hardware design knowledge was complex, the rapid pace of change made it impossible for a single engineer to know everything, thus favoring a specialist division of labor. This leveling of the playing field allowed even junior engineers to be teachers and demanded that even senior engineers become learners.

By contrast, structural engineering's slow rate of change meant that knowledge, once gained, was unlikely to become obsolete. Thus, accumulation of technical experience had value and allowed seniors to become generalists and masters whose knowledge of professional practice was broad and deep. These masters taught junior engineers, whose knowledge was narrow and shallow, as apprentices. Compared to their seniors, juniors had more to learn and, hence, were the most frequent learners. Because most of what juniors knew was already known to their seniors, most of juniors' teaching was confined to their own rank.

Markets. Although the slow pace of change allowed for generalism in structural engineering, it did not, however, require it. Many other fields in which knowledge changes slowly are marked by extreme specialization, as is the case among historians. What constrained structural engineering to generalism was the customized nature of their market. Buildings are built for particular clients who have particular needs that translate into an idiosyncratic set of building features (e.g., an open, airy lobby), components (e.g., trusses), and materials (e.g., wood). Because this set changed as the project progressed, structural engineers could not predict the precise knowledge they would need to complete the project at its start. Engineers assigned to the project, therefore, had to be able to handle the gamut of design possibilities, which meant that each engineer had to possess broad knowledge. By contrast, hardware engineering firms produced products for mass markets. Hardware engineers did not customize the product's features, components, and materials for the customer. Because each new version of a microprocessor required roughly the same kinds of knowledge as did the last, producing for a mass market over time did not contravene specialization.

Licensing. Licensing requirements were an additional environmental factor that led us to study structural and hardware engineering. California law specified that an engineer must have at least six years of experience "under the direction of a civil engineer legally qualified to practice" before sitting for the licensing exam (Professional Engineers Act 2003, §§6751–6752). The law further specified that the exam must test "knowledge of state laws, rules, and regulations, and of seismicity and structural engineering unique to practice in this state" (§6763.1). These licensing requirements provided a direct mandate for senior structural engineers to proactively teach juniors and to organize that learning around predetermined content. Licensing requirements help to explain why senior structural engineers took the time to tutor their juniors in general lessons when juniors asked problem-specific questions. In short, licensing requirements shaped the balance of seeking and offering as well as the teaching method in structural engineering. By contrast, in hardware engineering, licensing was an option but not a requirement. Thus, few hardware engineers sat for the state exam. Because there was no legal mandate for experienced hardware engineers to teach novices prescribed content, licensing had no material effect on how hardware engineers taught one another.

Competition. To understand why hardware engineers rarely offered to teach, one needs to consider a fourth environmental difference: the nature of competition in the market. Hardware engineers built products that competed with other firms' products. Because the duration of a firm's advantage as a first mover was short, designers worked under the threat that a competitor would beat them to market. One manager told us that on a recent project, "The engineers were working around the clock and sleeping in sleeping bags in their offices." Tight pace and high pressure meant that hardware engineers had little patience for instructing others. Hardware engineers were expected to respect others' time by learning on their own, if possible. Hardware engineers were only willing to teach when asked and largely confined their teaching to providing information. Structural engineering firms competed against each other by bidding on projects prior to design. Having eliminated their competition by winning a bid, structural engineers worked with deadlines and an established budget, but beyond the normal need for timely progress, there was no race to market. In sharp contrast to the hardware engineers, structural engineers routinely came to work at nine in the morning; in the afternoons, they left by six; almost none were set up with computers to work from home, and rarely did a project's budget permit overtime work. In short, the nature of competition in structural engineering did not channel the choice of teaching method or patterns of seeking and offering.

Judging the Soundness of Designs. The environments' constraints on teaching methods were reinforced by the differential ability of the engineers in the two occupations to judge the soundness of their designs, a difference rooted in their technologies. Structural engineers validated designs using commercial computer programs that assessed whether a design could withstand specified loads. The output of these programs depended on the soundness of the input, such as engineers' estimates of how loads would travel through the building. For this reason, programs could yield mathematically correct yet physically unsound results, as this senior engineer described:

> I reviewed a building where [a software program] had been used.... They sent me reams of computer printouts from it, but all I needed to do was look at the drawing. There was a place on the drawing where a beam was huge because the engineer said it was taking all the load because it had a fixed base. But beside it was this other beam and it was really skinny. And the thing is, the software kept telling the engineer to make the first beam bigger, and it kept taking more of the load, so it got bigger and bigger and this other one got really skinny. But that was all based on the assumption of the fixed base, and in reality it wasn't fixed at all! In fact,...all the load really had to go through that very skinny member. It all happened because some young kid out of college learned the software, and just did what the software told him to do.

Confirmation by the program alone was, therefore, insufficient. Complete validation required a careful examination of assumptions, which called for considerable engineering experience and judgment. Because such knowledge could not be easily learned on one's own or by a quick exchange of information, senior structural engineers taught juniors in a side-by-side, pedagogical fashion.

Assessing the soundness of a chip's design was much easier. Hardware engineers produced code that accepted predetermined input and produced predetermined output. Tests of the code were unambiguous: Either the code passed or failed. If the code passed, the designer knew his design was good. If the code failed, he knew he needed to locate and fix a bug. Rarely did debugging require tutoring in design principles or modeling assumptions. Rather, debugging usually required information about how the programs called by the designer's code operated. To gain this information, a designer could try to read other programs. However, if a program's documentation was incomplete or its logic was hard to follow, the designer's remaining option was to query the program's author. Thus, hardware engineers generally required specific pieces of information rather than side-by-side instruction on design techniques.

Discussion

Environmental Shaping of Structure via Action

Although organizational theorists have long accepted the idea that environments shape the structure of organizations, one of the field's longstanding dilemmas has been explaining how the shaping occurs without falling victim to either the Scylla of a mechanistic environmental determinism or the Charybdis of prescient management. Our analysis of the teaching-learning ecologies of engineering groups in three structural and three hardware engineering firms suggests a way to navigate between these two theoretical extremes. Like ethnomethodologists, students of social worlds, and students of structuration, we advocate a course that entails adopting the idea that no structure can exist unless it is instantiated in everyday life. Students of technology have shown that new technologies can operate as exogenous stimuli that shape behaviors that, in turn, reshape organizational structures (Barley 1990, Schultz and Orlikowski 2004). We submit that the attributes of an organization's environment operate similarly. Treating properties of the environment as stimuli for structuring brings action to center stage in the drama of organizing without needing to postulate heroic managers or institutional entrepreneurs who assess environments and then design organizations accordingly. Instead, what one needs to show is that environments generate a set of constraints and affordances that tend to channel behavior toward one pattern rather than another.

Showing that environments shape organizational structures through action requires researchers to adopt different tactics than those that students of technology and structuring have taken. Because adopting a new technology is usually planned, and because the implementation often unfolds over a matter of months or a few years, students of technology and structuring can study their phenomenon longitudinally in real time. Studying how environments shape action in real time is difficult because environmental changes (with the exception perhaps of changes in law) are rarely scheduled, and adaptation can span years, if not decades. Therefore, one approach to studying environmental shaping is to apply logic analogous to analysis of variance. This has been our approach.

We chose structural and hardware engineering because we knew that the rate of change of knowledge in the two fields was dramatically different. We also knew that, unlike hardware engineering, structural engineering was governed by strong licensing requirements. Our conjecture was that these environmental differences were likely to have broad implications for how engineers learn on the job. We choose engineering groups in three firms for each type of engineering guided by the premise that if the environment structures the daily activity of learning, then the teaching-learning ecologies of structural and hardware engineering would not only differ, but the ecology associated with each environment would also characterize each engineering group across firms operating in that environment. If so, then the data would be consistent with the logic of environmental structuring. As we have shown, this was indeed the case: The teaching-learning ecologies of structural engineering groups were similar across firms, the ecologies of the hardware engineering groups were similar across firms, and the ecologies of the two occupations differed from each other in ways that were consistent with the differences in their institutional and noninstitutional environments.

Although we selected the two engineering occupations based on differential rates of change in knowledge and the presence and absence of licensing requirements, our analysis suggests that these factors alone could not have accounted for the teaching-learning ecologies that we observed. Rapid changes in knowledge did constrain hardware engineers to be specialists, but slow change in knowledge did not constrain structural engineers to be generalists. Although slow change in knowledge did not prevent structural engineers from specializing, the custom market for buildings did. The custom market meant that each engineer had to be capable of handling whatever type of structure the architect and building owner chose in each project. Similarly, although licensing requirements pushed senior structural engineers to offer to teach in a pedagogical fashion, their absence did not push hardware engineers to teach primarily by providing information when asked. Had it not been for a competitive market and its associated time pressures, hardware engineers might well have taken the time to proactively tutor and teach more collaboratively.

Table 11 Factors That Shaped Teaching and Learning Behaviors

	Constraint?	
	Structural engineering	Hardware engineering
Environmental factors		
Rate of knowledge change in occupation	No	Yes
Nature of market	Yes	No
Licensing requirements	Yes	No
Nature of competition	No	Yes
Technological factor		
Ability to judge the soundness of designs	Yes	No

In other words, each of the environmental factors constrained activities in one occupation but not the other. As Table 11 shows, the nature of the market and licensing requirements constrained teaching and learning behaviors in structural but not hardware engineering. Conversely, the rate of knowledge change in the occupation and the nature of competition constrained behaviors in hardware but not structural engineering. Similarly, the ability to judge the soundness of designs constrained behaviors in structural but not hardware engineering. The two teaching-learning ecologies were, therefore, configured by unique profiles of constraints.

Note that the finding that only one extreme of an environmental factor may constrain organizing highlights an important but often misunderstood aspect of how original contingency theory portrayed environmental shaping. Contingency theorists differentiated the environment along continuous dimensions and associated specific organizational structures with each end of the continua. For example, mechanistic and organic structures were associated, respectively, with slow and fast technological change, with predictable and unpredictable markets, as well as with certain and uncertain technologies (Burns and Stalker 1961, Perrow 1967, Lawrence and Lorsch 1967). However, for early contingency theorists environmental factors were only constraining at one end of a continuum. For example, although an unpredictable environment might push organizations toward an organic structure, a predictable environment did not constrain firms to a mechanistic structure. Rather, predictability may have simply allowed firms to organize mechanistically, which they did for other reasons, such as the search for efficiency.

Individual Learning in Organizations

Although March's early research implied that individual learning is integral to the process by which organizations develop routines, practices, and structures, most recent research has focused on the dynamics of learning itself. For this reason, research on individual learning in organizations has spoken less directly than it might otherwise have to issues of broader concern to organization theory. By examining how environments channel the behaviors of teachers and learners into the patterned structures that we call teaching-learning ecologies, we illustrate how studies of learning may help resolve longstanding dilemmas in the field. Specifically, our analysis of learning activities provides a much-needed explanation for how environments can shape structures of organizations. In short, we advocate treating learning as one of the missing links between environment and structure.

Although at least three streams of research have conceptualized learning as a social activity, none of them simultaneously examined environment, structure, and action. To do so, we adopted a synthetic and pragmatic view of learning. By synthetic we mean that our approach combined features from all three streams. For example, we allowed that individuals might learn alone (as March conceived them), in dyads (as the helping and information-seeking literatures suggested), or in groups (as situated learning theorists have shown). We took from the helping and information-seeking literature the concept of a source, but acknowledged that sources teach. Doing so enabled us to bring into our analysis notions of teacher-initiated learning and differences in teaching methods. By pragmatic we mean that we construed learning in a way that allowed us to study learning as behavior as opposed to cognition. Accordingly, we restricted our inquiry to explicit knowledge, whose acquisition lent itself to empirical observation.

Taking this approach brought to the fore two aspects of learning in the workplace that have previously received relatively little attention. First, most prior research has underemphasized teaching. Teachers were absent from March and his colleagues' image of workplace learning because learning was social but not relational. Situated-learning theorists conceptualized knowledge as passing from old-timers to newcomers, but they downplayed explicit teaching and argued that most knowledge was transmitted subtly and unintentionally through newcomers' participation in community activities. The helping and information-seeking literatures placed sources of knowledge center stage, but the main intent of studies in these streams was to identify the networks of information transfer, not to distinguish among teaching methods and their use. Bringing teachers fully into the picture enabled us to examine differences in method by rank and occupation, which helped to characterize the distinctions in how learning unfolded across the two occupations. For example, in structural engineering, senior engineers frequently tutored, but midlevel and junior engineers did not. In hardware engineering no one tutored, and all ranks employed the same methods. Because such distinctions were so significant, we chose to speak of "teaching-learning" rather than simply "learning" ecologies.

Second, recognizing the role of teachers pushed us to realize that teachers could initiate learning episodes, a possibility that prior scholars have often overlooked. The extent to which individuals offered to teach and the rank of those who offered strongly differentiated the two ecologies: Offering occurred far more frequently in structural than in hardware engineering, and senior structural engineers offered to teach far more frequently than did their juniors. Our study suggests that paying attention to purposeful teaching may be a necessary condition for unearthing differences in how individual learning occurs across occupations and organizations.

Finally, because our research finds two distinct teaching-learning ecologies, it casts doubt on the existence of a universal model of learning. Consistent with situated-learning theory, individual learning in organizations does seem to be organized by communities of practice—in this case, occupations. However, situated-learning theory's imagery of knowledge passing from old-timers to newcomers proved viable only for structural engineering. In hardware engineering, seniors were less likely to teach than were midlevel engineers, who were, in turn, less likely to teach than junior engineers. Thus, the structure of teaching in hardware engineering was precisely the opposite of what situated-learning theory would predict. More broadly, if teaching-learning ecologies vary by occupation as our study suggests, then the number of distinct ecologies could be quite large. In fact, if occupations are the relevant unit of analysis for defining teaching-learning ecologies, then it is possible that many organizations contain multiple ecologies.

Overall, our study contributes to the literature on environmental shaping of structure by identifying in the context of learning how the everyday activities of all members of a social collective are crucial to shaping and maintenance of an organization's structure. Specifically, we showed how environmental factors such as rate of knowledge change, licensing requirements, nature of the market, and nature of competition shaped the actions of all individuals within the occupational groups we studied, and ultimately created structure. Our analysis returns organizational analysis to its microsocial foundations without relying on explanations that turn heavily on the actions of heroic individuals. Additionally, our study contributes to the literature on individual learning in organizations by showing that environments shape how learning unfolds within organizations. Whereas the helping and information-seeking literatures mapped who consulted whom at work, our study provides explanations for these patterns of teaching and learning that are rooted in environmental factors. We rule out notions of universal models of learning by documenting differences not only in patterns of who taught and who learned, but also in the methods by which teachers taught and the content of their teaching. Built upon Barker's work on streams of behavior, our detailed methods of observation and analysis provide a roadmap for future studies of teaching and learning behaviors in situ by demonstrating how to trace structures to sets of behaviors and actions. Future studies might apply these methods beyond the context of learning to investigate how other aspects of environments shape structure through other domains of action.

Acknowledgments
The authors gratefully acknowledge Julie Gainsburg and Lesley Sept, whose help on this project was instrumental in the field and in thinking about their data, and Mahesh Bhatia, Jan Chong, Fabrizio Ferraro, Menahem Gefen, and Carlos Rodriquez-Lluesma, who assisted in data collection. They thank the engineers for graciously allowing them to observe them at work and for sharing their thoughts with them. This research was made possible by National Science Foundation Grant IIS–0070468. The General Motors-Stanford Collaborative Research Laboratory provided additional funding.

Endnotes
[1]Because we oversampled junior and midlevel engineers relative to senior engineers, we had fewer opportunities to witness senior engineers in dyadic episodes, particularly those involving another senior engineer. To correct for this uneven distribution of observations and to allow us to make accurate comparisons across rank, we employed learning rates rather than raw counts of episodes. Interested readers may contact the authors for step-by-step explanation of how we calculated rates.

[2]This is not to say that for a particular attribute a given firm completely matched the pattern for its occupation; some variation certainly existed. However, no firm consistently violated the pattern for its occupation. For example, although dyadic episodes were most common across all three structural engineering firms, solo episodes were the next-most frequent in the firm that built fabrication plants for the computer chip industry, not group episodes as in the other two firms. Hardware firms showed similar variation, but as with structural engineering firms, the variation was not concentrated within a single firm.

References
Adler, P. S. 1993. The "learning bureaucracy:" New United Motor Manufacturing, Inc. B. M. Staw, L. L. Cummings, eds. *Research in Organizational Behavior*. JAI Press, Greenwich, CT, 111–194.

Allen, T. J. 1977. *Managing the Flow of Innovation*. MIT Press, Cambridge, MA.

Barker, R. G. 1963. *The Stream of Behavior*. Appleton-Century-Crofts, New York.

Barker, R. G., H. F. Wright. 1955. *Midwest and Its Children: The Psychological Ecology of an American Town*. Row, Peterson, Evanston, IL.

Barley, S. R. 1986. Technology as an occasion for structuring: Evidence from observations of CT scanners and the social order of radiology departments. *Admin. Sci. Quart.* **31**(1) 78–108.

Barley, S. R. 1988. The social construction of a machine: Ritual, superstition, magical thinking and other pragmatic responses to running a CT scanner. M. Locke, D. Gordon, eds. *Knowledge and Practice in Medicine: Social, Cultural and Historical Approaches*. Kluwer Academic Publishers, Boston, 497–540.

Barley, S. R. 1990. The alignment of technology and structure through roles and networks. *Admin. Sci. Quart.* **35**(1) 61–103.

Barley, S. R. 2005. What we know (and mostly don't know) about technical work. S. Ackroyd, R. Batt, P. Thompson, P. Tolbert, eds. *The Oxford Handbook of Work and Organization*. Oxford University Press, Oxford, UK, 376–403.

Barley, S. R. 2008. Coalface institutionalism. R. Greenwood, C. Oliver, R. Suddaby, K. Sahlin-Anderson, eds. *Handbook of Organizational Institutionalism*. Sage, Thousand Oaks, CA, 490–515.

Battilana, J. 2006. Agency and institutions: The enabling role of individuals' social position. *Organ.* **13**(5) 653–676.

Berger, P. L., T. Luckmann. 1967. *The Social Construction of Reality*. Doubleday, New York.

Borgatti, S. P., R. Cross. 2003. A relational view of information seeking and learning in social networks. *Management Sci.* **49**(4) 432–445.

Bourdieu, P. 1977. *Outline of a Theory of Practice*. Cambridge University Press, Cambridge, UK.

Brown, J. S., P. Duguid. 1991. Organizational learning and communities-of-practice: Toward a unified view of working, learning and innovation. *Organ. Sci.* **2**(1) 40–57.

Brown, J. S., P. Duguid. 2001. Knowledge and organization: A social-practice perspective. *Organ. Sci.* 1**2**(2) 198–213.

Burns, T. R., G. M. Stalker. 1961. *The Management of Innovation*. Tavistock Institute, London.

Burt, R. S. 2005. *Brokerage and Closure: An Introduction to Social Capital*. Oxford University Press, Oxford, UK.

Child, J. 1972. Organization structure, environment, and performance—The role of strategic choice. *Sociology* **6**(1) 1–22.

Cohen, M. D., J. G. March, J. P. Olsen. 1972. A garbage can model of organizational choice. *Admin. Sci. Quart.* **17**(1) 1–25.

Cook, S. D. N., J. S. Brown. 1999. Bridging epistemologies: The generative dance between organizational knowledge and organizational knowing. *Organ. Sci.* **10**(4) 381–400.

Cross, R., L. Sproull. 2004. More than an answer: Information relationships for actionable knowledge. *Organ. Sci.* **15**(4) 446–462.

Dewey, J. 1938. *Experience and Education*. Macmillan, New York.

DiMaggio, P. J., W.W. Powell. 1983. The iron cage revisited: Institutional isomorphism and collective rationality in organizational fields. *Amer. Soc. Rev.* **48**(2) 147–160.

Dobbin, F., J. R. Sutton, J. W. Meyer, W. R. Scott. 1993. Equal opportunity law and the construction of internal labor markets. *Amer. J. Soc.* **99**(4) 265–316.

Downey, G. L., A. Donovan, T. J. Elliott. 1989. The invisible engineer: How engineering ceased to be a problem in science and technology studies. L. L. Hargens, R. A. Jones, A. Pickering, eds. *Knowledge and Society: Studies in the Sociology of Science Past and Present*. JAI, Greenwich, CT, 189–216.

Edelman, L. B., M. C. Suchman. 1999. When the "haves" hold court: Speculations on the organizational internalization of law. *Law Soc. Rev.* **33**(4) 941–999.

Edmondson, A. C. 1999. Psychological safety and learning behavior in work teams. *Admin. Sci. Quart.* **44**(2) 350–383.

Feldman, M. S. 2004. Resources in emerging structures and processes of change. *Organ. Sci.* **15**(3) 295–309.

Feldman, M. S., B. T. Pentland. 2003. Reconceptualizing organizational routines as a source of flexibility and change. *Admin. Sci. Quart.* **48**(1) 94–118.

Galbraith, J. R. 1977. *Organizational Design*. Addison-Wesley, Reading, MA.

Garfinkel, H. 1967. *Studies in Ethnomethodology*. Prentice Hall, Englewood Cliffs, NJ.

Gherardi, S. 2006. *Organizational Knowledge: The Texture of Workplace Learning*. Blackwell, Maldern, MA.

Giddens, A. 1976. *New Rules of Sociological Method*. Hutchinson, London.

Giddens, A. 1984. *The Constitution of Society*. University of California Press, Berkeley.

Hannan, M. T., J. H. Freeman. 1977. The population ecology of organizations. *Amer. J. Soc.* **82**(5) 929–964.

Hannan, M. T., J. H. Freeman. 1988. *Organizational Ecology*. Harvard University Press, Cambridge, MA.

Henderson, K. 1999. *On Line and on Paper*. MIT Press, Cambridge, MA.

Heritage, J. 1993. *Garfinkel and Ethnomethodology*. Blackwell, London.

Hinds, P., S. Kiesler. 1995. Communication across boundaries: Work, structure, and use of communication technologies in a large organization. *Organ. Sci.* **6**(4) 373–393.

Hutchins, E. 1991. Organizing work by adaptation. *Organ. Sci.* **2**(1) 14–39.

Jordan, B. 1989. Cosmopolitical obstetrics: Some insights from the training of traditional midwives. *Soc. Sci. Med.* **28**(9) 925–937.

Kelly, E., F. Dobbin. 1999. Civil rights law and work: Sex discrimination and the rise of maternity leave policies. *Amer. J. Soc.* **105**(2) 455–492.

Kogut, B., U. Zander. 1992. Knowledge of the firm, combinative capabilities, and the replication of technology. *Organ. Sci.* **3**(3) 383–397.

Krackhardt, D. 1994. Graph theoretical dimensions of informal organizations. K. Carley, M. Prietula, eds. *Computational Organizational Theory*. Lawrence Erlbaum Associates, Hillsdale, NJ, 89–111.

Lave, J., E. Wenger. 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press, New York.

Lawrence, P. R., J. W. Lorsch. 1967. *Organization and Environment: Managing Differentiation and Integration*. Richard D. Irwin, Homewood, IL.

Lawrence, T. B., N. Phillips. 2004. From *Moby Dick* to *Free Willy*: Macro-cultural discourse and institutional entrepreneurship in emerging institutional fields. *Organization* **11**(5) 1350–1384.

Lazega, E., C. Lemercier, L. Mounier. 2006. A spinning top model of formal organization and informal behavior: Dynamics of advice networks among judges in a commercial court. *Eur. Management Rev.* **3**(2) 113–122.

Levitt, B., J. G. March. 1988. Organizational learning. *Annual Rev. Soc.* **14** 319–340.

March, J. G., J. P. Olsen. 1975. The uncertainty of the past: Organizational learning under ambiguity. *Eur. J. Political Res.* **3**(2) 147–171.

March, J. G., J. P. Olsen. 1976. *Ambiguity and Choice in Organizations*. Universitetforlaget, Bergen, Norway.

Meyer, J. W., B. Rowan. 1977. Institutionalized organizations: Formal structure as myth and ceremony. *Amer. J. Soc.* **83**(2) 340–363.

Meyer, J. W., W. R. Scott, T. E. Deal. 1981. Institutional and technical sources of organizational structure. H. D. Stein, ed. *Organization and the Human Services*. Temple University Press, Philadelphia, 151–178.

Morrison, E. W. 1993. Newcomer information seeking: Exploring types, modes, sources, and outcomes. *Acad. Management J.* **36**(3) 557–589.

Orlikowski, W. J. 1992. The duality of technology: Rethinking the concept of technology in organizations. *Organ. Sci.* **3**(3) 398–427.

Orr, J. E. 1996. *Talking About Machines: An Ethnography of a Modern Job*. ILR Press, Ithaca, NY.

Perlow, L. A. 1997. *Finding Time*. ILR Press, Ithaca, NY.

Perlow, L. A. 2001. Time to coordinate: Toward an understanding of work-time standards and norms in a multicountry study of software engineers. *Work Occup.* **28**(1) 91–111.

Perlow, L. A., J. Weeks. 2002. Who's helping whom? Layers of culture and workplace behavior. *J. Organ. Behav.* **23**(4) 354–361.

Perlow, L. A., J. H. Gittell, N. Katz. 2004. Contextualizing patterns of work group interaction: Toward a nested theory of structuration. *Organ. Sci.* **15**(5) 520–536.

Perrow, C. 1967. A framework for the comparative analysis of organizations. *Amer. Soc. Rev.* **32**(2) 194–208.

Professional Engineers Act. 2003. Business and Professions Code Sections 6700–6799, California Law, http://www.leginfo.ca.gov/calaw.html.

Powell, W. W. 1990. Neither market nor hierarchy: Network forms of organization. B. M. Staw, L. L. Cummings, eds. *Research in Organizational Behavior*. JAI Press, Greenwich, CT, 295–335.

Powell, W. W., J. A. Colyvas. 2008. Microfoundations of institutional theory. R. Greenwood, C. Oliver, R. Suddaby, K. Sahlin-Anderson, eds. *Handbook of Organizational Institutionalism*. Sage, Thousand Oaks, CA, 490–515.

Schoggen, M., L. S. Barker, R. G. Barker. 1963. Structure of behavior of American and English children. R. G. Barker, ed. *The Stream of Behavior: Exploration of Its Structure and Content*. Appleton-Century-Crofts, New York, 160–168.

Schoggen, P. 1989. *Behavior Settings: A Revision and Extension of Roger G. Barker's Ecological Psychology*. Stanford University Press, Stanford, CA.

Schultze, U., W. J. Orlikowski. 2004. A practice perspective on technology-mediated network relations: The use of Internet-based self-serve technologies. *Inform. Systems Res.* **15**(1) 87–106.

Sole, D., A. Edmondson. 2002. Situated knowledge and learning in dispersed teams. *Brit. J. Management* **13**(2) S17–S34.

Suchman, L. A. 1987. *Plans and Situated Action: The Problem of Human-Machine Communication*. Cambridge University Press, Cambridge, UK.

Tolbert, P. S., L. G. Zucker. 1983. Institutional sources of change in the formal structure of organizations: The diffusion of civil service reform, 1880–1935. *Admin. Sci. Quart.* **28**(1) 22–39.

Vaughan, D. 1996. *The Challenger Launch Decision: Risky Technology, Culture and Deviance at NASA*. University of Chicago Press, Chicago.

Vincenti, W. G. 1990. *What Engineers Know and How They Know It*. Johns Hopkins University Press, Baltimore.

Wenger, E. 1998. *Communities of Practice: Learning, Meaning and Identity*. Cambridge University Press, Cambridge, UK.

Woodward, J. 1958. *Management and Technology*. HMSO, London.

Yanow, D. 2003. Seeing organizational learning: A cultural view. D. Nicolini, S. Gherardi, D. Yannow, eds. *Knowing in Organizations: A Practice-Based Approach*. M.E. Sharpe, Armonk, NY, 32–52.

Zucker, L. G. 1977. The role of institutionalization in cultural persistence. *Amer. Soc. Rev.* **42**(5) 726–743.

Diane E. Bailey is an assistant professor in the School of Information at the University of Texas at Austin. Her most recent research investigates the work practices associated with, and the impact of, sophisticated computational and logic computer applications in engineering design and analysis. Other research interests include telework, knowledge, learning, and teams.

Stephen R. Barley is a professor of management science and engineering and the codirector of the Center for Work, Technology, and Organization at Stanford's School of Engineering. He received his Ph.D. from MIT in organization studies. His research interests include work and occupations, technological change, and corporate power in representative democracies.