Career paths of non-engineers into engineering practice in the midst of globalization: Implications for engineering education

Juan C. Lucena
Division of Liberal Arts and International Studies
Colorado School of Mines, Golden, 80401
Phone: (303) 273-3991; e-mail: jlucena@mines.edu

Abstract

Corporations, governments, and the employees they hire, face increasing challenges of the global economy such as mobility of capital and labor, organizational re-structuring across national boundaries, development and implementation of more efficient production and manufacturing practices. Yet we know very little about how engineers understand and experience globalization, and how globalization impacts their education, practices, and collaborations with non-engineers. For example, organizational changes and initiatives implemented to respond to global competition, such as mergers, joint ventures, product customization, subcontracting, etc, create circumstances for non-engineers to be hired as engineers as actually practice engineering. Under these circumstances, non-engineers' knowledges and skills become important for the solution of technical problems. Collected through ethnographic fieldwork and interviews, the data presented here show how processes of globalization open engineering practice to the application of non-engineering knowledge by non-engineers to complex technical problems. The analysis of this data has a number of implications for engineering education for it shows to non-engineers alternative career paths into engineering, reveals the value of non-engineering knowledge and skills in the solution of technical problems, and sheds light into the limitations of the educational engineering pipeline as a metaphor of engineering education.

Introduction

In the United States, most corporate employers of engineers require a bachelor's degree in engineering from ABET accredited institution for entry-level engineering positions. However, the reality of engineering employment significantly deviates from these requirements. According to the National Science Foundation (NSF), in 1995 of the total 1.6 million people employed in engineering occupations there were 400,000 people (or 25%) with no engineering degrees. "While people without engineering degrees are found in all engineering occupations, the largest share is in the rapidly growing and vaguely defined occupation of computer software engineering; these engineers have degrees in all fields-including the humanities. Computer hardware engineers without engineering degrees often have degrees in the physical sciences." (Burton and Parker 1998) By 1999, the share of non-engineers working as engineers still remains at 25% (National Science Foundation 1999). The percentages of engineering practitioners without an engineering degree varies significantly according to occupation: 7% in civil engineering, 11% in mechanical engineering, 13% in electrical engineering, 25% in biomedical engineering, 40% in computer hardware engineering, and 60% in software engineering. (Burton and Parker 1998).

The NSF reports from where this data was taken provide no plausible reasons for the presence of non-engineers in engineering practice. One can begin to look at examples where non-engineers are being hired into engineering practice to get a sense for the reasons behind this phenomenon. For example, in response to the deficiencies in management education and skills of engineers, engineering societies, and even large public infrastructure organizations like the California Department of Transportation (Calstrans), have opened their engineering management ranks to non-engineers. (Bellinger 1995; 1996)To compensate for shortages of engineers, Reliance Industries Limited (RIL), India's largest petrochemical and exporter (RIL produces 3% of India's GDP), has made a pact with the Indian Institute of Technology (IIT) of Powai for the upgrading of skills of its non-engineering employees (1997). These two examples point to lack of management skills in engineers and shortage of engineers as reasons to incorporate non-engineers into the ranks of engineering.

However, I argue that besides labor shortages and deficiencies in engineering education (e.g., lack of management education in the engineering curriculum) we also need to consider how corporate practices, resulting from the challenges of global competition, might open engineering practice to non-engineers. The data and analysis presented here comes from fieldwork and interviews with two non-engineers in software engineering, the engineering occupation with the highest percentage of non-engineering practitioners. For the last 20 years there has been an ongoing debate on whether software engineering deserves to be included in the disciplines of engineering. (Diaz-Herrera 2001; Modesitt, Bagert et al. 2001; Rhodes 2002) Sorting out this debate is beyond the scope of this paper. However, I argue that the corporate practices presented here present a challenge to all engineering disciplines and occupations since they are not unique to one company, industry or country.

Since in order to understand why and how non-engineers end up working as engineers we need to go beyond labor shortage data, the analysis below tries to answer the following questions: What changes in the workplace, resulting from increasing globalization, might be opening engineering practice to non-engineers? How might these changes make non-engineering-knowledge and skills appropriate for the solution of complex technical problems? While attempting to answer these questions might involve lengthy and expensive longitudinal studies of non-engineers' careers, we can begin to shed light to these questions and attempt to formulate better ones for further research by exploring the careers of two non-engineers using interviews and ethnographic field work.

Methodology

The experiences of non-engineers in engineering practice presented here come from ethnographic fieldwork and interviews conducted as part of a NSF-sponsored four-year multisite ethnography in major aerospace corporations, engineering education programs, professional meetings, and other aerospace sites. At each site, ethnographic research included background research of engineering activities, archival research, in-depth interviewing with a diversity of engineers, and participant observation during seminars, workshops, courses and meetings. Data has been collected in the form of field notes, recorded interviews, pictures, literature, and hand-notes and drawings from engineers. Recorded interviews have been transcribed into text files, coded, and analyzed using NVIVO, a qualitative analysis software package. The full names and corporate affiliations of the interviewees have been concealed to protect their identity, as agreed in the informed consent form signed by all participants.

Ethnographic data provides a thick description of people's experiences that cannot be captured by using quantitative methods. Ethnography tries to bring to light the ways in which a group of people experience their lives by "focus[ing] on the way words and other cultural constructs with which people conceive of and talk about their world affect their experiences." (Hakken and Andrews 1993) Data from ethnographic research is helpful in establishing patterns, discovering what might be plausible under different circumstances, and giving voice to otherwise hidden experiences, as it is the case of non-engineers in engineering. However, ethnography coexists uneasily with generalizations (Geertz 1973; Kunda 1992). Hence I am not trying to provide

generalizations about non-engineers' experiences in engineering practice. At this point, the literature review reveals that no one is analyzing the experiences and career paths of non-engineers practicing engineering. So what I am attempting here is to provide plausible patterns of the experiences of non-engineers involved in engineering practice with the hope to establish a larger research agenda that quantitative scholars interested in engineering knowledge and practice might want to pursue.

Results

Background: How non-engineers enter engineering practice

Pat graduated with a biology degree from a small liberal arts college in Ohio in 1975. The next year she obtained a job with the U.S. Department of Agriculture in a Southwestern state where she collected data on cotton plants while someone else entered the data in a computer database. When the data entry person left, Pat took the job without any prior knowledge of computers. Immediately she began to take FORTRAN and COBOL classes at a nearby community college. Realizing that there were not many well-paid jobs for biologists, she found a job as a subcontractor "database modeler" for the US Air Force. After one year of building visual databases for flight simulators, Pat began software development for the flight simulator and continued taking computer classes. After 6 months as a subcontractor, she began to work directly for the US Air Force. In the mid-1980's Pat's supervisor moved from the US Air Force to a private aerospace company that was competing for the US Army LHX helicopter. The LHX was a R&D project sponsored by the US Army to develop a reconnaissance and attack helicopter capable of flying at high speeds. After losing the LHX bid in 1991, Pat's company had to layoff

many engineers. Pat did not get laid off and soon moved on to another military helicopter project. By now she had taught herself ADA computer language and began writing flight simulator software. By the time of the interviews, Pat had become an "embedded software engineer" at a major aerospace company. She is the technical lead for the systems processor of a very sophisticated military helicopter. In Pat's own words, "I'm a technical lead for that processor. Which means I get to make decisions nobody else makes. Or nobody else wants to make. And get blamed if stuff doesn't work quite right…Basically, I just design based upon requirements from the aviation group, design the software, code it, test it, put it in the box." (Ohio 2000)

Bob is a member of the Church of Latter Day Saints. After his high school graduation, he went on a mission to Spanish-speaking areas of California. Wanting to get into teaching--both his parents were teachers-- and highly influenced by his father's profession --Bob's father was a ranger and owned a pet shop-- Bob decided to major in botany with three minors in zoology, English, and Spanish. He took many courses in math, life sciences, English, Spanish, drama: "I am a generalist, I do like everything," said Bob when explaining the whole range of subjects that he took in college. (Schoenhals 2001) Upon graduating from college, Bob moved with his wife and child to Vermont to get a job teaching high school science earning $7,000 a year. After a while, wanting to be close to their families in Utah and California, Bob and his family moved to California. According to Bob, there were no available teaching jobs in Utah and California at the time so he decided to try for a job in the computer industry. He was hired by Sperry-Univac, a company that at the time was competing in the computer market with the UNIVAC 1106 and 1108 and its operating system EXEC 8. Since Bob had no prior knowledge of computers, he had to teach

himself computer programming. Moving to another location in California to work for the same company in the US Marines air traffic control system, Bob found difficulties in understanding new computer languages so he took classes at nearby community colleges. Bob built his skills as a programmer to the point that he made the two companies working in the air traffic control system compete for his skills. Bob went to the company that offered him a higher salary. Due to the failed attempt to rescue the hostages in Iran, this project lost its funding. Bob was laid off but now felt confident enough about his computer programming skills to go to high-tech job fair where he competed with other engineers for jobs in sophisticated weapons systems. Bob was hired as en engineer. Not liking the prospect of having to send their teenage children to high school in California, Bob and his wife decided to move out of the state. After two major aircraft companies merged and decided to place their military helicopter facility out of state, many California engineers decided not to follow their company. Bob was hired as an engineer to work on processors. By the time of the interviews, Bob had become a "software engineer" in a major aerospace company. He works writing communication software for a display processor of a military helicopter for which he writes software for a high frequency (HF) radio that could transmit over the horizon, making the radio system installed on the helicopter be compatible with ground systems.

Non-engineering knowledge and skills to solve technical problems

Both Pat and Bob brought their own disciplinary knowledge and skills to the solution of technical problems. When asked to describe her favorite subject, Pat began describing ecology as follows:

> Um, it's more how if you do something to one part in the ecosystem, it affects something else. Like, you hear stories about how, you know, there's some rodent or something somebody wants to get rid of, and they say, oh, in this country, you know, this animal eats

> up those rodents. So, they bring that animal in and they overpopulate, and it causes another problem. It's the whole, how everything is interrelated. And you can't really change one piece of the puzzle without affecting everything else. (Ohio 2000)

As a software developer for the US Air Force, Pat explained how her knowledge of ecology facilitated the way she developed visual databases. Through ecology Pat came to understand that "you can't really change one piece of the puzzle without affecting everything else." Applying this lesson to a new technical problem –the limitations of the system—Pat said:

> Well the systems weren't good enough for that. You basically had, you know, some almost triangular thing, which was a mountain, and then you just had, like rectangles or something would be roads and rivers and things. So, the systems at the time weren't good enough for that. But, we used to joke about playing God. You know, saying we don't like where a mountain is so we're going to move it someplace else… It wasn't God in the sense that we were very powerful, it was just God in the sense of we could imagine what we wanted the landscape to look like for the airplane to fly through. And a lot of the work was basically working around system limitations. There were several different system limitations that were sometimes at odds with each other and you had to figure out how to get, how you could see the most and get around all the problem and the limitations of the system. (Ohio 2000)

Pat explains the similarities and differences between software engineering and ecology. In software engineering, you take all the different pieces and then make them fit all together in one big picture. In ecology you begin with the big picture and then figure out how the smaller parts fit, live, or are connected in this big picture. However, for her, engineering databases is similar to ecology because different areas of the database affect one another once they become integrated in the simulator.

> Pat: Well, it's problem solving and it's taking different aspects of, in that case the image generator, and understanding how they all fit together, and come up with a solution. It was kind of the opposite of what I said before about ecology. But it's sort of the same thing because if I built a database in a certain way that I don't exceed the polygon limitation I may exceed some other limitation, and you still end up with a screwed up system. So you kind of have to work one area but make sure you don't adversely affect another. So in that respect they're sort of the same.(Ohio 2000)

When hired at Sperry-Univac, Bob had no prior knowledge of computers so he had to teach himself computer programming. He learned computer languages in the same way the he learned foreign languages: teaching himself by reading books, asking people, and paying attention to the relationship between words as in crossword puzzles.

> Bob: We were working for a West German contract, their district marking thing for air traffic control and I learned Jovial language. Basically, after a few weeks I went home to my wife and said, "I can't believe this. They are playing me to play games," basically, it was fun. The logic, it was like paying me to do crossword puzzles. I couldn't believe it, it was so neat, I had thought I would hate it. But after a while I said, "This is great!" …They basically gave you books, gave you a job, and you could ask people, what the heck is this and what is binary and I studied hex and I don't know, it was self taught, there were no classes at that time. The reason they said they hired me was because there were not enough people graduating from any computer science courses in 1976, there were a few but the aerospace industry was going into computers big time and there weren't enough people so they just hired anybody and if they could do it ok. So, anyway I did that for about three years. (Schoenhals 2001)

Bob reflected on how his easiness with foreign languages played a big role in the way he learned computer languages:

> Maybe that helped because I also studied German. That is one of my side things is languages. I speak Dutch, Spanish, a little German, Hebrew, English that is my hobby, one of my hobbies. Maybe that helped because, you know, language is language and I just, I took to Jovial really well, I liked it. I was able to produce code and test it and it worked and documentation was great because I was, I knew English, I spoke fairly good English, I hope that is grammatically correct. (Schoenhals 2001)

As we can see, our non-engineers bring knowledge and skills from their own educational backgrounds to solve technical problems. Interesting research questions for further study emerge from these two experiences: What kind of tensions, conflicts, and/or resolutions emerge between non-engineering and engineering knowledge and skills in the solution of technical problems? How

does the transfer of non-engineering knowledge and skills to engineers (and engineering knowledge and skills to non-engineers) take place? Can we teach our engineering and non-engineering students to deal with these tensions and conflicts and to learn from each other?

Corporate competitiveness opens engineering practice to non-engineers

Scholars of engineering have documented ways in which an increase in economic competitiveness, beginning in the 1980s and accelerated with the globalization of markets from the 1990s to the present, has shaped engineering practice and education. (Kunda 1992; Downey 1998; Lucena 2000) However, analysis of the relationship between competitiveness and the participation of non-engineers in engineering are lacking. The experiences of our two non-engineers reveal ways in which corporate competitiveness might provide opportunities for non-engineers to apply their knowledge to different technical problems.

For Pat, competition between two corporate consortia for the LHX helicopter design created an opportunity to apply ecology to develop visual databases, first for the LHX and then for a new military attack helicopter. After losing the LHX bid in 1991, Pat's company had to layoff as many as 1,000 people (Luebke 1991). Pat did not get laid off and soon moved on to another military helicopter project. By now she had taught herself ADA computer language and began writing flight simulator software. Pat explained that the difference between her previous job as database modeler and her new job as software developer lies in who sets the design requirements. From her knowledge of ecology, she knows that she needs to understand the big picture of how everything fits together:

> Well, if you're writing software, basically, you have to have a good understanding of what you want the software to do, and then it's just a matter of writing code to make it do what you want it to do. When I was in database group I decided what I wanted it to do, I had

> to use my background in database to do that, to come up with the requirements. And then I wrote the code to meet those requirements. In [this new helicopter project] software avionics comes up with the requirements. They specify in detail what the system is supposed to do, and then I either, if it's a modification to existing code then I just modify the existing code to do that. If it's a new function, then basically I just write new software to meet those requirements. So, and it's the same language. We started out on the same kind of computer; we've since transitioned to another kind so… So, considering the fact that I didn't know anything about the [helicopter] or any of its subsystems, it was a fairly easy transition…And I 'm still learning how the big picture of how everything fits together. But that's a life-long job I figure. (Ohio 2000)

In Bob's case, Sperry-Univac's competition with IBM, Honeywell/GE, RCA, and Burroughs in 1970s created an opportunity for his foreign language skills to be transformed into computer language skills. Moving to another location in California to work for the same company in the US Marines air traffic control system, known as CMS2M, Bob found difficulties in understanding new computer languages so he took classes at nearby community colleges.

Bob built his skills as a programmer to the point that he made the two companies working in the air traffic control system compete for his skills. Bob went to the company that offered him a higher salary. Due to the failed attempt to rescue the hostages in Iran, the US Marines air traffic control project lost its funding. Bob was laid off but now felt confident enough about his ability to learn new computer languages to go to high-tech job fair where he competed with other engineers for jobs in sophisticated weapons systems. His skills of translating how he learned foreign languages into learning computer languages had become one of his most important engineering skills:

> Bob: So, they [Sperry] laid us off. So, I went to a job fair and I got employed by ITT, but it was Federal Electric (FEC) which was a subsidiary of ITT at Vandenburg Air force Base and there I started working on what was called range safety. Whenever we had rocket launches and we were doing the minuteman, MX missile that was it. Which was one of the first big multiple warhead rockets, it would launch ten different war heads and hit precision, ten different sites in a certain area, that kind of thing. So, we would launch to Quaduling, which was an island in the Pacific and I did a lot of the report generation

> from all of the sensors, it would gather all this radar information and all this stuff and in order for people to read it you had to go through all the data and put it into columns and stuff…And that is where I learned FORTRAN and, once again, you get on the job and they give you a book and they tell you you are going to work in FORTRAN and they give you a book and, basically, you look for the same things. The names are all different and they had fields and jovial and they had, I forget all the names but every language has its own words. And they describe in software all these things you use, records, files, data structure, all this stuff, it is really….I continually studied language. I would always listen to Spanish; still I listen to Spanish on the way on in… So language is just natural and computers it is the same thing. I mean this means that and this means that. I don't know, maybe, you might want to pick a correlation, but me I just loved coding and documentation, I have to do it. (Schoenhals 2001)

As we can see, our non-engineers found employment opportunities in engineering practice in a world of corporate competitiveness. But many research questions remain. For example, what is the relationship between number of non-engineers working as engineers and corporate competitiveness by sector? Are the most competitive companies attracting more non-engineers to work as engineers? What kind of non-engineering knowledge and skills have higher demand in different competitive sectors and companies? What are the consequences of these answers to engineering education?

Product customization provides non-engineers with plenty of work

Product customization, a strategy of corporate competitiveness, brings continuous change in systems integration and generates enough work to create demand for the engineering skills of our two non-engineers. Pat finds herself continuously writing software for the systems processor of the military helicopter as new customers make new equipment requirements. As technical lead for software development for the systems processor, Pat has to constantly coordinate changes in systems requirements. Trying to maintain military advantage over other countries, the US Army demands new batches of military helicopters with the latest hardware and software every six

years. At the same time, armies from other countries that purchase this helicopter order new equipment like radios, weapons, etc. to be installed in the helicopter. Every new requirement requires new software to be written and documented. Pat stays busy.

Juan: What's the latest called?

Pat: Well, they're all [new models]; but every year there's a new lot put out. Almost every lot has got software changes, and sometimes hardware changes associated with it. So depending upon what lot [number] is, it's different from the other lots. So, Multi Year I was Lots 1-6, Multi Year II is Lots 7 through something. So, we continually improve it. So, in that respect Multi Year II isn't any different from Multi Year I.

Juan: And is this, are you working on projects that are for the U.S. Army, or are these for projects that…?

Pat: This is for the U.S. Army, but what we've got working, I think December 1st is going to be the baseline for the Singapore Apaches. And it's…

Juan: December 1? What do you mean the baseline?

Pat: It's the starting point. So, by December 1st we have to be converted to Ada 95 and running on a Power PC. And that's what the Singapore Apaches are going to start with, and then they're going to make their own changes from there.

Juan: Do you see yourself in the future, or are you going to see yourself working directly for the Singapore Apaches, or it doesn't make a difference? Help me understand how they…

Pat: In terms of writing code it really doesn't make much difference, but I suspect I'll be staying on U.S. and other people will be assigned to Singapore.

Juan: And how does that, does that make any difference to your work or how you see your work?

Pat: Um, in terms of what I'm doing now, well, it makes it more important that our code is obviously correct, easy to maintain. Because if we write some code that's a real pain to maintain then it's going to be a pain to maintain for multiple projects, not just…It would be one thing if I was writing some code and I did it in a way that I'd understand but nobody else, and that might be one thing if I was the only one maintaining the code. But our code just spreads out among all the countries so, really need to do as good a job as possible. (Ohio 2000)

Similarly, Bob finds plenty of work as he shifts between testing, teaching, writing software for different requirements for the display processor of the military helicopter.

> Bob: I specifically, pretty much, have confined myself to the communications devices and every country that signs up with us wants another different radio and that is a lot of work, it is just a lot of work. You have to put the radio on, you have to communicate with it, it has all these counter measure and its got modems and so. Basically, one of the biggest changes every time we have a new contract, like I worked on the Netherlands, that one was one of my big projects, the Dutch helicopter. We have another group in the U.K.; the British have their version. We are doing Singapore, United Arab Emirates; Israel just signed a new contract to upgrade to the [helicopter]. So, we have the main U.S. Army software suite, which we call, we go in lots. Lots one, two, three, we are up to four and they are planning five, six, seven, eight, down the, you know, that always adds new capabilities and new software and new hardware. But breaking off from that we will often have other countries that want differences from the U.S. Army or are not allowed to have certain things that the U.S. Army has; countermeasures and codes and so forth. (Schoenhals 2001)

As we have seen, our non-engineers continue to find engineering work in the midst of product customization. Here as well interesting research questions for further study emerge. As product customization increases, as more countries are expected to buy high tech military systems previously unavailable to them, should we expect more non-engineers to be involved as engineers in the design, development, and maintenance of customized products? Are there any substantial differences between the process of product customization that involves non-engineers as engineers and that which does not? What are the consequences of these answers for engineering education?

Conclusion: Implications for engineering education

The experiences of our two non-engineers reveal not only interesting research questions in engineering practice but also raise significant issues for engineering education, particularly for

problem-solving –as one of the cornerstones of engineering education—and the engineering pipeline –as the dominant metaphor of engineering education.

Problem solving. These two stories reveal how non-engineers bring their disciplinary knowledge and skills to solve technical problems in engineering practice. From ecology, Pat brings a holistic approach to software design. Robert applies his method for learning foreign languages to his learning of computer languages so he can write code. When Pat and Robert work alongside other engineers and non-engineers, we can safely assume that different practitioners of engineering (engineers and non-engineers alike) bring different perspectives to problem solving. Well known figures in the world of engineering have made strong calls for the importance of diversity of perspectives for engineering design and practice (Wulf 1998) yet engineering education still provides students with one dominant perspective to problem solving: textbook problem solving. In this perspective, engineering students are always given problem, learn how to extract relevant information from the problem statement, draw a free body diagram to visualize the direction and magnitude of forces, identify the scientific principles that apply to this situation, work through the math, and produce one solution. All major publishers of engineering textbooks have introductory textbooks on engineering problem solving (Eide and et.al. 1998; Burghardt 1999) and most subsequent engineering science textbooks continue to reinforce the method. Students are constantly rewarded for the proper use of this method, and the one right solution, throughout most of their engineering curriculum. By the time they are graduating seniors they have come to accept this method as the cornerstone of engineering. We have argued elsewhere that resistance to design education might come from students' sense that what counts as engineering is the

problem-solving method learned in their engineering sciences courses (Downey and Lucena 2001). However, as corporate competitiveness, and its associated practices like product customization, open engineering practice to more non-engineers, we should expect more engineers to be solving problems alongside non-engineers. Hence we need engineering students who can value, appreciate, and, probably most important, learn other approaches to problem solving. The new realities of engineering practice, and the reasonable expectation that globalization processes will continue to open engineering practice to non-engineers, should motivate engineering educators to think of innovative ways to educate future engineers to understand and value non-engineering problem solving perspectives.

Limitations of the pipeline as a metaphor of engineering education. The experiences of our two non-engineers also shed light into the limitations of the educational engineering pipeline as a metaphor of education that assumes linear career paths from K to employment. The pipeline metaphor requires us to focus on recruitment and retention, to recognize "leaks" or "drop-outs" as people who have little, if any, chance to come back into the pipeline, and even to ignore non-engineers who might want to come into the pipeline at later stages. Explaining the limitations of the pipeline, a former senior education- and human-resources policy advisor to the National Science Board said in an interview

> the pipeline metaphor is not a very useful metaphor...because it restricts at various entry points and there aren't too many of them. There are still more people flowing out than people flowing in. It restricts the pool of people who could eventually go on and take degrees in science and engineering. So if you fix the pipeline, if you were to seal the leaks, you are still dealing with a very small minority of all the students. (Chubin 1994)

Also aware of the limitations of the pipeline, an NSF deputy division director in engineering

education argued:

> There is one big flaw of the pipeline....There are a lot of people who do not go on a linear path through school, especially after grade 10. I know an American Indian woman who drops out of school in 10th grade and has finally come back at age 35 and gets a GED and goes to the community college and at age 40 has an associate degree and by age 43 has a bachelor's degree...we have more and more of those kind of people...If you look at the average student in the California State University System or the Texas A&M system, they are not 18-year-old white men. The average profile is that of a woman who is 25-35 and that doesn't jive with the pipeline analogy. That is a big problem for science and engineering. (Kemnitzer 1994)

The stories of our non-engineers clearly show that we need a new metaphor for engineering education and employment, one that helps us understand the career paths of those who become engineers after, and even without, traditional undergraduate engineering education and that allows us to view every non-engineer major as a potential engineer later. Activists for diversity in science and engineering education have begun to question the usefulness of the pipeline metaphor (Bordogna 2002; Muller and Metz 2002a) and to propose alternatives like that of "a river with feeders, streams, tributaries, oxbows, deep pools and shallow riffles, etc." which better reflects the complexity of pathways in and out of engineering education and employment. (Muller and Metz 2002b) The engineering education community needs to address this challenge by formulating recruitment and retention policies and programs based on more appropriate metaphors that reflect the complexities of the engineering workforce, especially when globalization processes will continue to open the paths of engineering practice to non-engineers.

References

1996. Caltrans may fill 2 top posts with non-engineers. Los Angeles Daily News.

1997. Extended Title: Reliance Industries Ltd to upgrade the skills of its non-engineering employees through an agreement with the Indian Institute of Technology. Financial Express.

Bellinger, Robert. 1995. IEEE looks outside the ranks for new GM. Electronic Engineering Times. 77.

Bordogna, Joseph. 2002. *From Pipelines to Pathways*. Assessing the Impact: ATE National Principal Investigators Conference,

Burghardt, M.D. 1999. *Introduction to Engineering Design and Problem Solving*. New York: McGraw Hill.

Burton, Lawrence and Linda Parker. 1998. Degrees and occupations in engineering: How much do they diverge? Arlington, VA, National Science Foundation.

Chubin, D. 1994. Interview with author.Washington, D.C,21 November.

Diaz-Herrera, J.L. 2001. *Engineering design for software: on defining the software engineering profession*. Frontiers in Education Conference, 2001.,

Downey, Gary. 1998. *The Machine in Me: An Anthropologists Sits Among Computer Engineers*. New York: Routledge.

Downey, Gary and Juan C. Lucena. 2001. When Students Resist: An Ethnography of Design Education in Engineering.In *Mudd Design Workshop III: Social Dimensions of Engineering Design*. C. L. Dym and L. Winner,281-292. Claremont, CA: Harvey Mudd College.

Eide, A.R and et.al. 1998. *Introduction to Engineering Problem Solving*. New York: McGraw Hill.

Geertz, C. 1973. *The Interpretation of Culture*. New York: Basic Books Press.

Hakken, David and Barbara Andrews. 1993. *Computing Myths, Class Realities: An Ethnography of Technology and Working People in Sheffield, England*. Oxford: Westview Press.

Kemnitzer, S. 1994. Interview with author.Washington, D.C.,27 October.

Kunda, Gideon. 1992. *Engineering Culture: Control and Commitment in a High-Tech Corporation*. Philadelphia: Temple University Press.
copyright: Temple Univ Press

Lucena, Juan. 2000. Making Women and Minorities in Science and Engineering: Nation, NSF, and Policy for Statistical Categories. *Journal of Women and Minorities in Science and Engineering*

Luebke, Cathy. 1991. Valley defense industry hit hard by budget crunch. The Business Journal - Serving Phoenix & the Valley of the Sun. 1(2).

Modesitt, K.L., D. Bagert, et al. 2001. *Academic software engineering: what is and what could be? Results of the first annual survey for international SE programs*. ICSE 2001. Proceedings of the 23rd International Conference on Software Engineering,

Muller, C. and S. Metz. 2002a. Burying the Pipeline and Opening Avenues to Engineering. ASEE Prism. **12:** 72.

Muller, C. and S. Metz. 2002b. E-mail communication with the author.11 December.

National Science Foundation, Division of Science Resources Studies. 1999. Scientists and Engineers Statistical Data System (SESTAT). Division of Science Resources Studies.

Ohio, P.[pseud.]. 2000. Interview with author.31 March and 25 September.

Rhodes, D.H. 2002. *Systems engineering: an essential engineering discipline for the 21st century*. ICSE 2002. Proceedings of the 24rd International Conference onSoftware Engineering,

Schoenhals, Bob [pseud.]. 2001. Interview with author.31 March, 2000, and 26 March, 2001.

Wulf, W.A. 1998. Diversity in Engineering. *The Bridge* 28(4):

JUAN LUCENA is Director of the McBride Honors Program in Public Affairs for Engineers and Associate Professor at the Liberal Arts and International Studies Division (LAIS) at the Colorado School of Mines. Juan obtained a Ph.D. in Science and Technology Studies (STS) from Virginia Tech and a MS in STS and BS in Mechanical and Aeronautical Engineering from Rensselaer. Currently, he is researching how images of globalization shape engineering education, hiring practices, and engineering practices and designs under a NSF CAREER Award titled *Global Engineers: An Ethnography of Globalization in the Education, Hiring Practices and Designs of Engineers in Europe, Latin America, and the U.S.* He is also a co-PI on a project to develop, evaluate, and disseminate curricula on the cultural dimensions of engineering education and practice.