CHEMICAL ENGINEERING - A CENTENNIAL CELEBRATION

Koffolt Labs

1903-2003

THE OHIO STATE UNIVERSITY

Unsolved Problems in Process/Product Systems Engineering

Biographical sketch of Arthur W. Westerberg

Arthur W. Westerberg is currently the Swearingen University Professor of Chemical Engineering at Carnegie Mellon. He received his BS from Minnesota '60, MS from Princeton '61 and PhD from Imperial College, University of London '64, all in chemical engineering. He spent two years at Control Data Corporation and nine years as a faculty member at the University of Florida before joining Carnegie Mellon University in 1976. He has served as Director of the Design Research Center (78-80), Chemical Engineering Department Head (80-83), and founding Director of the Engineering Design Research Center, an NSF funded Engineering Research Center (86-89).

His research interests are in engineering design, specifically in design synthesis, analysis, optimization, computer support environments for modeling and design collaboration.

A member of the National Academy of Engineering, he is the recipient of several awards from AIChE, ASEE, the ACS and CMU.

Unsolved Problems in Process/Product Systems Engineering

Arthur W. Westerberg
Dept. of Chemical Engineering and
the Institute for Complex Engineered Systems
Carnegie Mellon University
Pittsburgh, PA 15213
aw0a@andrew.cmu.edu

Abstract

We examine the process/product systems engineering (PSE) area of chemical engineering. A brief history of some of the accomplishments of the past four decades follows and notes the strong coupling of PSE with developments in computing. "Driving forces" in place today suggest where PSE research and development might move. The paper ends with a discussion of some major outstanding problems in PSE.

Introduction

Today the Chemical Engineering Department of Ohio State University is celebrating its 100th year, a very proud accomplishment. As a part of that celebration the department has invited several faculty from throughout the country to prepare papers and presentations, each of whose theme is to be the unsolved problems in one of the research areas in this discipline. Mine is to examine the relatively young area of process systems engineering. I will take the liberty of adding "product" to the title as we are more and more interested in delivering processes and products. While young, this area has been active, and, in the conventional process area, it is reaching maturity. Maturity means there are many others in the area, and all are competing to find and develop new ideas and to obtain research funds. With maturity comes the obligation to re-examine accomplishments and to think about where we should be heading. In this manner we are in the same situation that many believe chemical engineering as a discipline is in at this time. In spite of our maturity, some very difficult issues remain unresolved. We shall examine them in the final section of this paper.

What is process systems engineering?

Three years ago, Ignacio Grossmann and I prepared a "Perspectives in Chemical Engineering" article for the AIChE J on the process systems engineering area[1]. Our first task was to define this area. We noted "companies must design and operate chemical processes effectively and efficiently so they may survive in today's highly competitive world. Providing the methods, tools and people that allow industry to meet its needs by tying science to engineering is a compelling aspect of Process Systems Engineering (PSE)." I find it hard to improve on this view.

We elected to broaden the more conventional view that our domain was process design, control and operation. We start at the nano-scale with decisions about what molecules to make. At the next scale, we often have to consider the aggregation and structuring of materials from these molecules. These first two types of decision are more in line with thinking about products rather than processes. We progress to our traditional scale, where we make decisions about production units and then whole processes. Companies worry about the proper designing and operating of multiple processes. Finally, at the meso-scale, companies, interlinked into supply chains, make decisions about how they fit into the whole of the manufacturing and delivery activity for an industry. Our move to the meso-scale is coming quickly. For example, in January of next year in Kunming, China, we have organized the seventh conference on PSE around the topic of business decision making (http://pse2003.chemeng.tsinghua.edu.cn/).

Our final definition for PSE became: "*Process Systems Engineering is concerned with the improvement of decision making processes for the creation and operation of the chemical supply chain. It deals with the discovery, design, manufacture and distribution of chemical products in the context of many conflicting goals.*"

What is different about PSE?

We noted in our perspectives paper that much of chemical engineering research is science-based. Its intellectual base is the study of nature, where the goal is to seek and understand the "truths" of nature. We perform as engineers when we use this understanding to create useful artifacts for the betterment of humanity. PSE resides within the activity of engineering, and its intellectual challenge is the discovery of concepts and models that will allow us to improve all levels of decision making processes. What we create does not have to be an exact representation but rather a useful one that leads us with least effort to correct decisions.

What are some accomplishments in PSE?

Computing

PSE as an active research area started in the late 1950s, at the same time that engineers could access computers that were more than adding machines. The first computer I used was a Univac 1101. It was a room full of tubes (valves in UK English) that would have, had the air conditioning failed, led to the burning down of the building in which it was housed. It had 1000 words of memory and a computer language that was one to one with machine instructions – and it was little more than programming in 0s and 1s. By 1960, I was using a dual drum IBM 650 at the Forrestal Research Center at Princeton with twice 2048 words of rotating drum memory and by the next spring a CDC 1604, a transistor-based machine with 32K words (at this time words were of the order of four bytes). The first supercomputer was likely the CDC 6600 that appeared in the early 1960s. In 1977 we saw the CRAY 1, a super-supercomputer that, according to the folklore I have heard, had about the capability of a 100 megahertz Pentium. The IBM personal computer appeared in 1983 – I had two. It had 640K words of fast memory and a 10 megabyte disk and cost about $4400. 1995 was the year of the internet explosion. Today we have giga-everything on our desk. My latest computer is dual multithreaded xeons rated at 2.66 gigahertz each, with 2 gigabytes of fast memory and a dual, auto backup, 200 gigabyte disk. My new Ethernet card has a gigabit performance label.

Analysis and optimization

With this development in computing, PSE continually attacked larger and larger problems. Three data points are the solving of 19 equations in 19 unknowns in about as many hours in the late 1950s, to the bragging about solving 650 equations in the mid-1960s, to the solving of a million equations today in a few tens of minutes.

In the late 1960s, we could optimize a flowsheet, while using the computer time required to perform about 2000 flowsheet simulations. It took the entire weekend and often failed to get to the answer. People became skilled at saving intermediate results every hour so they would not have to start over when the inevitable failure occurred. With the advent of the generalized reduced gradient algorithm, optimizing a flowsheet used the equivalent in time of 100 converged flowsheet calculations. In the late 1970s, we participated in the development of the sequential quadratic programming algorithm, which uses a Newton-based scheme to solve the model equations and the necessary equations for optimality together. We demonstrated that we could optimize in about the time we took simply to solve the flowsheet equations alone. We believe it cannot get much better. There has been a lot of later work to improve one's ability to converge to solutions under adverse numerical conditions, and throughout the 80s and 90s, we participated in the creation of systems to aid engineers in writing and debugging process models.

Discrete decision-making

In the early 1980s, Ignacio Grossmann added binary variables to our models. His first work showed how to set up and solve mixed integer linear programming (MILP) problems. Binary variables allow one to choose among discrete alternatives while optimizing, such as choosing between using a single- or a two-stage compressor or the choosing of the number of trays in a column. The solving algorithms involve the use of decision trees and bounds rather than just gradients, and they take a lot longer. He and his students subsequently developed the DICCOPT algorithm, which became a part of the commercially available modeling system GAMS and thus accessible to all of us to solve mixed integer nonlinear programming (MINLP) problems. Many in PSE have now contributed to the literature that abounds with many, many tricks in formulating and strategizing about classes of problems so one can solve them. Industry often attests to using this type of method and saving millions of dollars annually. One example is to plan where and when to place oil rigs in the Caribbean. Another is to plan the investments for annual plant modifications while simultaneously making monthly production decisions to control inventories when one can only guess future demand and technology improvement. These techniques are routinely use to schedule batch/semi-continuous, multi-purpose plants.

Synthesis

Another area in which PSE researchers contributed is in developing so-called "process synthesis" methods to aid engineers to be creative when designing new processes. A very early paper on distillation by Lockhart in 1947 gave guidelines on whether to separate an ideal ternary mixture by separating the most or the least volatile component first. In the late 1960s, Dale Rudd and his students really started others to look at this area. They asked if one could automate the selection of equipment and its configuration first for heat exchanger networks and then for a complete flowsheet. In 1971 Ed Hohmann, a student of Lockhart, showed in his PhD thesis that one could compute the least amount of hot and cold utilities required for a process WITHOUT knowing the heat exchanger network that could accomplish it. He tried to publish twice, and peer review rejected his papers. He also showed how one could estimate the heat exchange area required. Linnhoff in 1977 independently discovered the same ideas using a different formulation and brought these ideas into engineering practice. With these tools, synthesis reached the goal of outperforming what engineers could do without them.

In the 1970s, Gary Powers and many in chemistry looked at doing what I remember doing as a student in organic chemistry in the late 1950s: invent different ways to synthesize a particular organic molecule. This automation showed that there were literally millions of pathways, most of which unaided humans would never think to explore.

In addition, the early 1970s produced many papers on how to find the best sequence of distillation columns when separating relatively ideal mixtures. The real fun was the work on designing distillation and decanter systems for separating highly non-ideal mixtures, where azeotropes and liquid/liquid behavior abound. This latter work appeared first in the late 60s in the ex-Soviet literature (they had to be smart on paper because they did not have computers – see a soon-to-appear book by Petlyuk) and then with the work of Doherty and Perkins and then Doherty and his students.

In the early 1980s, Tennessee Eastman (now Eastman Chemicals) built a plant to manufacture methyl acetate that required about one sixth (yes, 85% less) the investment and operating expenditures of previous plants. Their design had one extractive, reactive distillation column in it, and it started many research efforts into understanding reactive distillation. Again, among the first were Doherty and his students. Several of the simulation companies added reaction to their column simulation packages. However, simulation and much of the analysis did not suggest to anyone if one should use reaction and, if so, where to place it in a column. Work still progresses in this area, and we now have a number of insights that can help engineers be creative in the design of these systems.

Other areas

Most of the above discussion relates to the design of conventional chemical processes. Major activities in PSE also cover the control and operation of chemical processes. Process control has moved from the using of frequency response methods and simple PID controllers in the late 50s, to computer supported optimal control in the 60s, to heavily computer supported model predictive control in the last two decades and lately to less complex but guaranteed stable control based on passivity theory. Better analysis methods include creating more accurate dynamic simulators, creating very fast simulators for training, showing how to analyze, optimize and control dynamic and distributed systems, and providing methods to determine the stability of a controlled process. Synthesis methods include aiding engineers to decide what to measure, how to structure the control system, and how to alter the design to improve one's ability to control a process.

We also see work, often based on artificial intelligence approaches, for creating computer support systems to help operators, and now computers, determine what just went wrong and how to take safe corrective action in the running of a process.

Control systems involve the interaction of operators, computer programs, PLC circuits, control elements, and the process units. There is much opportunity in such a complex structure for there to be errors, many of which could be hidden catastrophes in the making. Can one hope to "verify" the correctness of such systems? Gary Powers, continuing his earlier work on safety, introduced the use of methods to verify the correctness of computer chips (e.g, a Pentium computer chip) to checking such systems. He is talking about 10^{50} pathways, which makes Avogadro's number looks small in comparison.

So what is next?

PSE has had a rich and exciting history. The academic community has also grown from only a handful of us in the 1960s to hundreds today (with a large percentage -- often stated to be about half of the academics -- on the Roger Sargent family tree). Engineers now routinely use our tools and methods to improve how they design, control, and operate processes.

Our domain for the last four decades has been chemical processes, with an emphasis on commodity chemical processes. While far from a fully explored problem area, it is becoming increasingly difficult to use problems from this domain to raise research funds. If we mention the word "distillation," support agencies decline our proposals almost out of hand. So are we becoming a research topic without a problem domain? I will suggest that the future of PSE is going to be in many, very diverse research areas that the about-to-be-discussed drivers suggest. This moving of PSE research into other domains is already happening. We cannot just move, however; we must become domain experts in these new areas to feel comfortable in contributing to them. I see an analogy to many in computer science who have proposed and developed excellent new approaches to problem solving but have lacked interesting problems on which to apply them. To me it has been much more comfortable to be an engineer who then learns computer science than the reverse.

But what will be the domains into which we should migrate or areas out of which future PSE researchers will migrate into PSE? Let us explore the answer by first looking at the world today to identify the "drivers" that will push or pull us into what we will do. The following are my thoughts.

Drivers for research

Let us assume (though the last two years could make this a dubious assumption) that people have increasing amounts of "extra" funds, that is, funds over and above the amount they deem necessary to survive. On what will we spend our extra funds without questioning the price? I would suggest two: on our **health** and on our **leisure**. On most other things, we tend to question price, but, on these two, we seem to accept the price charged. If so, these two areas may be where business will put its money in the future. We should at least review what we propose against this contention.

We are increasingly **socially aware**. We worry a lot more about harming the environment, about sustainability and about the total life cycle costs of decisions. This worry must increase as we continue to harm our planet more and more. Thus, it will have to be a driver for the future.

We are learning how to manufacture very small things, with the dimensions of molecules. NSF has programs almost requiring us to say the word "**nano**" in the title (with the consequence that a number of proposals now discuss 1300 nanometer sized objects). Materials where we can maneuver atoms into desirable positions have truly remarkable properties. They are often remarkably stronger, or they are able to detect and handle minute amounts of material, and so forth. We will be able to do for pennies what now cost hundreds of dollars – and we can now make these nano-objects. We are only scratching the surface. So "tiny" is a driving force of the future. Related to this nano area and by using thinner and thinner lines, computers are still following Moore's law. They will ultimately run into the limits of physics, but not yet. Perhaps we have another decade. What can we hope to do with major computing that is tiny, fast and virtually without cost for the hardware?

But in computing, it is the **network** that is likely the real contributor to our future. We are fast becoming a wired planet of people. There are two implications: information is unbelievably available, and we can work with people around the globe almost as easily as with someone in the next office. For information, how many of us hesitate now to look up any topic using Google, and with amazing success? One can even use it to check the spelling of a word (unless it is commonly misspelled). When we want to know about a company – we type in www.companyname.com and find everything we would ever want to know. We want to buy a digital camera, we find reviews everywhere and get back lists of vendors sorted by price.

Networking is creating another driver: **globalization**. Companies now operate around the globe. We can no longer operate in isolation. Globalization gives us more markets, but it also gives us more competitors.

Another driver is our progress in **biology**. Our ability to know, to interpret and to modify gene sequences is changing such things as medicine, our understanding of biology, and how we will grow our food.

To support life as we know it, governments build or regulate the building of **infrastructures** such as transportation systems, power generation and distribution systems, gas pipeline systems, road systems, phone systems, water distribution systems, waste recovery systems and so forth. Many of the tools we use in PSE are directly applicable for the design and operation of infrastructures. Deregulating many of our infrastructures has caused turmoil; this will continue as we explore more and more how best to manage these large, often monopolistic systems. New technology is constantly altering the rules for how to construct them also, as with cellular phone systems and power generation and distribution systems. Integrating them, such as where the transportation system uses the phone system to effect control, is impacting these systems.

These systems are becoming more integrated and much more complex. So handling **complexity** is a driver, too.

And yes, there are still new algorithms to discover and tools to build

For about four decades now, we have seen enormous improvements in the speed of computer hardware. There have been equally impressive improvements in algorithms, many of which we just discussed and that we in PSE have contributed, such as when solving the difficult problems we like to pose.

The improvements typically come from tailoring algorithms to fit the particular features of the problem. The first method to solve simultaneous nonlinear equations involved gridding the space and writing embedded DO loops. The approach found the best point in the grid and around it regridded with a finer mesh. Next, we used iterative methods, such as successive substitution, in our flowsheet systems. We added accelerators like Wegstein's method and Broyden's method. Then we implemented full Gauss elimination algorithms as the inner loop of Newton-Raphson methods, but we could only deal with a few tens of equations. Sparse matrix methods developed in modeling electrical circuits migrated into chemical engineering in the 1970s and dramatically improved computation times for the linear equation solving inner loop of Newton-Raphson based methods. We also learned how and when to use iterative methods for solving linear equations for extra large problems. Optimization went from pattern search to GRG to SQP, as noted earlier. We have contributed codes for solving mixed integer nonlinear programming problems that are widely used. We have also explored parallel computing whereby we decompose problems into parts that we solve in parallel.

It is evident in all this progress that many of us have been tool builders. Can we continue to be tool builders? Actually, I like the analogy to a tree falling in the forest. If no one is there to hear it, was there a sound? If no one uses our methods, was there a method? Yes, we will have to continue to build tools, but fewer of us will get credit for their novelty.

Is there anything new out there that will change the effectiveness of computing? Let us assume that we can have unlimited amounts of high speed networked computers for solving our problems. Can we find effective ways to exploit such a resource to solve very large hard problems? Parallel computing is a possibility. Unfortunately, in virtually all decompositions to parallelize algorithms, one finds a coordination step for which all the parallel steps must wait before carrying out the next iteration. If coordination takes ten percent of the time, then Amdahl's law states we are limited to a 10 fold decrease in "clock" time, even if we have hundreds of computers.

Agent-based systems: Another approach is the research of John Siirola, who is currently doing his PhD under the direction of Steinar Hauan and me. His topic is the use of agent-based computing to solve hard problems. This type of approach has been in the literature for about two decades, but we have not been exploiting it in chemical engineering. People have generally only used the method without examining quantitatively its effectiveness.

We can use the following metaphor to appreciate the ideas here. Suppose you were to direct a new research program at the NSF and were given a budget of \$10 million per year. You could choose to fund the one best, perhaps best by far, researcher in the US, or you could choose to fund the top two dozen. Most of us would agree that the latter approach is better. Now hold back 5% of the budget and insist that these researchers meet twice a year and share results.

There are two features of this last approach that appeal. We should like the diversity of having two dozen research programs looking at the topic. Second, the collaboration should aid researchers to see in the results of others better ways to proceed in their own program. Agent-based software systems have both these features. John has measured the impact of both diversity and collaboration, and both dramatically improve performance on a contrived but very difficult optimization problem. With five degrees of freedom, his problem has 10^8 local optima and with ten over 10^{17} (10^{17} microseconds equals 3200 years). He has also investigated having two competing objectives where his goal is to find the trade-off curve (Pareto surface) for them. Single agents seldom find the global optimum; diverse agent systems typically do, and they find better answers remarkably faster. In parallel computing ten computers might allow problem solution to be four times faster than for one computer; in agent-based systems the time reduction could be by a factor of 100.

We see in this approach the opportunity to alter how we solve hard problems. It is a "softer" approach. There are a few elusive proofs already available, but when tested this approach really works. It has a "social" model underlying it, and we know this cannot be science – right? But it could open some new research areas or allow us to revisit some old ones we could not tackle before.

Unsolved problems

I will illustrate the kinds of problems that seem to me to be unsolved in the process systems area. The first example will take care of problems being too large. Then we will consider problems that we do not know how to solve yet, although many people have worked at solving small examples of them.

I cannot solve a problem with 10,000 binary variables – yet

A trivial answer for noting an unsolved problem is to say that we always want to solve modeling, optimization, control and operating problems that are larger than our computers will allow us to solve at this time. The implicit solution is that we need only wait, and larger computers will solve the problem. History predicts we will solve problems about 10,000 times larger in the next decade, and there are 10,000 seconds in a week. However, we should question if it is only the need for larger computers.

This simple three column flowsheet will not converge

In analysis and optimization, we all often fail to converge our problems. Much work in convergence has to do with improving the numerical schemes. For example, there are modified Newton schemes that take bounded steps between a steepest descent and a Newton direction. These really do make a difference, but I am a firm believer that we need physical insights to converge most of our problems. For example, we may well place a specification on a separation device that device cannot deliver. In azeotropic separation, we may initialize a column in the wrong “distillation region” from which it will fail to converge, while initializing in the right region, even if done poorly, will generally succeed. In a “four sided” distillation region, we can find our distillation problem to be so sensitive to the numbers (changing the least significant bit of a double precision number) that the solution will jump from one behavior to a very different one. It is still an unsolved problem as to how we can provide feedback that tells an engineer why his problem is not converging and how he can overcome any such problem. We have made inroads on this problem in our ASCEND system, but we can only provide “clues” that the engineer still must decipher. I believe the approach will be to find hidden embedded near singularities and bifurcations and to trace their meaning through the problem. Connecting such a trace to its physical meaning will be an enormous benefit, but it will be very difficult.

I need a good solvent

A “cute” way to think of a reverse problem is that it is one in which we supply the answer and ask for the question. Reverse questions are at the heart of product and process design, and thus especially of process systems engineering. Assume our tools are models and not experiments, a good assumption for PSE. Given what we know and can model, what answers can we find that will aid us in engineering a solution to a problem? The flowsheet synthesis problem is a reverse problem of discovering a flowsheet that can manufacture a desired product. Another is to find a good catalyst using computational methods. Reverse questions always have many, many candidate solutions.

There are a few simple reverse questions that we keep thinking we should be able to solve, and we still cannot do them very well. One is to find components, mixtures of components and/or structures of components that have some desired physical properties. This type of problem is at the heart of designing a chemical product. For example, we use the Wilson method to predict the vapor-liquid behavior of many common organic mixtures. Suppose that we supply the physical behavior we want – e.g., we want a new solvent for our process. What solvent should we use? There is a considerable literature on this simple problem. Most of the time a paper on this topic will suggest stating the activity coefficient one wants and then seeking the groups (as in group contribution methods) that will give such a coefficient. The problem is then one of constructing molecules from those groups, and these papers typically find there to be many, including some good surprises. Unfortunately, only a few of the components we wish to consider have group contribution methods available. Also what makes a good solvent is a lot more than an activity coefficient. How much it costs, its toxicity, how easy it is to handle, can we recover it, and so forth are all part of the evaluation. Also, when one thinks of “order of magnitude,” the number of components in our libraries for which group contribution methods work is not that much larger than the number of groups. With gigahertz computers, we can search over a lot of components directly. So we can often try all pure component possibilities rather easily. Mixtures and structures are another story. There will be mixed integer programs formulated to solve these problems. What we suffer from more than anything though is a lack of good models for predicting the properties we want. That puts this problem in part outside the PSE area.

Should I buy this tanker of crude oil?

I pick this problem as it is a very large and very critical one for an oil company, and it illustrates the complexity of making business decisions. Buying a wrong tanker of crude can have a serious economic impact on a company, much more than solving its design, control, and operating problems optimally. The problem is many faceted. First, how does the company estimate with any accuracy what is in the tanker? Can its refineries process it? Oil companies use information on the oil field from which it comes as a first indicator. Several measurements the suppliers made on the crude oil aid the oil company to estimate what it is it might be purchasing. However, getting representative samples on which to make the measurements is a difficult problem. The simple question as to how much oil is actually in the tanker is also surprisingly difficult to answer, even if one monitors the delivery. Suppose the company has a good estimate of what is in the tanker. It next needs to account for its forecasts as to what it needs to deliver as products for the next few months. What is currently in its storage tanks? What products can it make from these inventories when it considers having this tanker of crude and when it does not? It will have to anticipate what crude it can buy in the near future, also. Where are available tanks located into which it can unload this tanker? What is in them? Will it mix or not? Should it? Is there a “slot” available at the unloading facilities? Perhaps the company should wait for the next tanker of crude? How will currency fluctuations affect this decision? Should it buy the tanker with the intention to trade it in a day or so?

Can we hope to create and maintain the models needed to respond to all these questions? One will need to represent many of the variables probabilistically. Can we solve such an immense problem? What is the right problem to solve? Are we interested in the low risk solution more than the high profit one? How will our decision affect our competitor's position?

Problem formulation – the oil tanker problem, second chapter

We already have many tools for analysis and optimization. As the above problem should have made clear, we now suffer from not knowing what problems we need to solve. Formulation is more important than solving. If we formulate the wrong problems, we get answers experienced engineers will reject out of hand. While it may sound trite, it is better to solve the right problem approximately than the wrong one optimally.

To understand how formulation is critical, what are the main selling points for portable computers today? Computer speed is not everything. Rather customers are also very interested in battery life, reliability, weight, screen resolution, security, connectivity, ruggedness, and service. If one overlooks an important customer need, one could doom his product.

Can there be strategies we can uncover to aid in formulation? Can there be tools we can develop to do the same?

How can I get my company to be one that survives? (subtitled: "Designing as a research topic") – and the saga started above continues

I also believe that designing is an important research domain but designing in a much broader sense than we have generally defined it to date. There is no doubt that the tools we discussed above are enablers for a company to be good at designing. Companies interested in low cost have to be skilled at modeling and optimizing. Companies should not ignore the available synthesis tools whose positive impact many have now well documented.

These tools enable a company to be good, but they do not make a company great. We suggest that only the great companies are likely to be here in the future. What makes a company great? Is there research open to chemical engineers that speaks to this question? I suggest that the great companies are those that understand what they are about, are creative, and understand what their customers want.

In commodity chemicals, price is likely the major issue. But is it? An anecdote will illustrate. A to-be unnamed, but real, chemical company delivered a commodity chemical to one of its customers. This supplier company discovered it could deliver a product that went from being 98% pure to being 99% pure while decreasing its costs. Of course, a purer feed should please its customer. Well, it did not. The customer had tuned its process to use the less pure feedstock. Its product became slightly discolored. It almost switched suppliers until the supplier company reverted to its old way of running its process.

Another anecdote is more personal. In 1986 we won NSF funding for our Engineering Design Research Center. Our goal was to improve the competitiveness of US industry by improving how it did its designing. We argued that by bringing many engineering and other disciplines together, we would greatly accelerate the sharing of methods and tools. We envisioned supplying much improved computer-based analysis and optimization tools.

Westinghouse brought us one of our first projects. We were to watch them design a new control system for a 2000 megawatt coal fired power plant, one in which the improved control would reduce emissions while not impacting overall efficiencies of the plant. We quickly discovered that they seldom were running computer simulations and optimizations. Other researchers doing ethnographic studies of such activities have estimated, as did we, that the designers used these tools for only about 15% of the time they spent on the design. Rather the design team members were in meetings sharing information, discussing how to make decisions, what decisions they had made and so forth. We started very quickly to question how we were to improve how they would do design if we only considered supplying improved analysis and optimization tools.

Since the spring of 1999, I have been directing the teaching of a course on product design at CMU. It is open to all juniors, seniors and graduate students at CMU. Company sponsors supply us with problems and expertise, as well as funding, to run the course. We have discovered that diversity on these teams (having a computer scientist, a chemical engineer, an industrial designer, a mechanical engineer and a business major on the same team) dramatically improves what they will create, something the students always find very surprising and enjoyable.

A main theme of a very interesting book by Bucciarelli[2] is that designing is a social as well as a technical process. Design teams carry out designs, and the design it selects, of the many possible, is a strong function of the personal values of its members. He argues that design is social because of the process by which one carries it out and social by the impact that personal values have on its outcome. To quote Bucciarelli: "The realization that design is a social process, that alternative designs are possible, and that a design's quality is as much a question of culture and context as it is of a thing in itself or of the dictates of science or market forces -- all this is a prerequisite to moving beyond simplistic images and myths about technology and doing better as designers, as corporate strategists, as government regulators, as consumers, and as citizens."

This topic is fraught with problems. The most significant is that humans and their behavior are a major part of what we must understand. Chemical engineering researchers are usually not very interested in such a soft problem. However, how can we really aid designers if we ignore these issues? Only companies that deal with these issues will be great at designing.

We have found our research niche in this area by developing tools to support information management and to support collaboration. This area is far from understood. Can chemical engineering researchers contribute to it? Should they? Obviously I have thought so and still do. I do not see how we can forsake understanding what it takes to make companies great rather than just good.

References

[1] Grossmann, I.E., and A.W. Westerberg, "Research Challenges in Process Systems Engineering," Invited Perspective, AIChE J, 46(9), 1700-1703 (2000).

[2] Bucciarelli, L, Designing Engineers , MIT Press (1994).