Undergraduate Computational Science and Engineering Education

SIAM Working Group on CSE Undergraduate Education,
Peter Turner, Chair
Contact address: Mathematics & Computer Science
Clarkson University, Potsdam NY 13699-5815
pturner@clarkson.edu

September 20, 2006

1 Introduction

In many areas of science and engineering, computation has become an equal and indispensable partner, along with theory and experiment, in the quest for knowledge and the advancement of technology. Numerical simulation enables the study of complex systems and natural phenomena that would be too expensive or dangerous, or even impossible, to study by direct experimentation. An increase during the past 30 years of over six orders of magnitude in computer speed, and another six orders of magnitude in algorithm speed, along with advances in mathematics in understanding and modeling complex systems, and in computer science of manipulating and visualizing large amounts of data, has enabled computational scientists and engineers to solve large-scale problems that were once thought intractable.

Why should you introduce an undergraduate CSE program at your school? There are at least four fundamental reasons. All of these will be expanded upon in subsequent sections of this report.

1. Computational science and engineering (CSE) is a rapidly growing multidisciplinary area with connections to the sciences, engineering, mathematics and computer science. CSE focuses on the development of problem-solving methodologies and robust tools for the solution of scientific and engineering problems. We believe that CSE will play an important if not dominating role for the future of the scientific discovery process and engineering design.

2. It is widely documented that the number and proportion of female undergraduates in computing fields has been declining over recent years. CSE, and especially CSE applied to biological sciences, typically attract a much

higher proportion of females. It is not uncommon for undergraduate applied mathematics programs to have a majority of female students, and it very common for biology, for example. CSE therefore represents a good opportunity to attract a more diverse student body into computing.

3. Also well documented is the shortage of good teachers in the K-12 system in the STEM (science, technology, engineering and mathematics) disciplines. Graduates trained in CSE offer a chance to seed the K-12 system with good mathematics and science teachers who understand the importance of applications and the power of computing to "real life."

4. For all the above reasons there is increased funding available for projects aimed at enhancing the CSE educational experience at the undergraduate level, and for applied and computational outreach to the K-12 community.

Throughout the remainder we will use the generic CSE to include alternative titles such as computational science, computational engineering, or other variations.

It is natural that SIAM, as the society whose aim is to foster the computational and applied mathematics, which is at the core of CSE, should play a role in the growth and development of this new discipline. The SIAM Working Group on CSE Graduate Education was formed in November 1998 to study the recent developments in CSE Education and to give recommendations regarding SIAM's potential role. That effort resulted in the report [1]. A substantial number of successful graduate programs in CSE have by now been established [14].

The SIAM Working Group on CSE Undergraduate Education was formed in February 2005 to report on the status of CSE undergraduate education, including what we have and what we need. The group was initially comprised of Peter Turner (Clarkson University), Linda Petzold (UC Santa Barbara), Angela Shiflet (Wofford College), Ignatios Vakalis (California Polytechnic and State University), and Kirk Jordan (IBM). The objectives of this report are to attempt to outline the scope of CSE as an undergraduate discipline, to examine some of the different models for CSE undergraduate programs and present some case studies, to delineate the needs that undergraduate CSE preparation must address for a successful transition either directly to industry or to CSE graduate programs, and to profile some recent graduates of CSE undergraduate programs and the careers they have chosen.

This report is a response to several years of rapid growth in CSE at all levels, and in undergraduate education in particular. The graduate report [1] was first presented in a special session at the International Congress on Industrial and Applied Mathematics in Washington DC in July 1991. The following year the first SIAM Conference on CSE was held and it included the first sessions devoted to undergraduate CSE, a minisymposium organized by Kris Stewart (San Diego State) and Ignatios Vakalis and contributed papers. From that point there has been a steady build up of interest.

SIAM's SIAG on CSE has rapidly grown to become the largest special interest group within SIAM, and to quote from the SIAG's conference website for CSE07:

> Computational Science and Engineering (CS&E) is now widely accepted, along with theory and experiment, as a crucial third mode of scientific investigation and engineering design. Aerospace, automotive, biological, chemical, semiconductor, and other industrial sectors now rely on simulation for technical decision support. For government agencies also, CS&E has become an essential support for decisions on resources, transportation, and defense. Finally, in many new areas such as medicine, the life sciences, management and marketing (e.g. data- and stream mining), and finance, techniques and algorithms from CS&E are of growing importance.

Undergraduate programs and education in CSE have shared in this expansion of interest with a steady growth in the number of sessions at both CSE meetings and SIAM National meetings. At SIAM's National meeting in Montreal in 2003, there was an evening Town Hall meeting attended by close to 50 delegates discussing the nature of CSE undergraduate education. That meeting also saw the first session for undergraduate research papers in Computational Science and Engineering.

By February of 2005, at the SIAM Conference on CSE, CSE05, there were multiple sessions on programs, courses, tools, assessment, outreach to the K-12 community and other topics. The undergraduate paper sessions were moved from the National Meeting to the CSE meeting and this resulted in three sessions of undergraduate papers with authors from 6 different countries.

In response to the obvious interest from the community, SIAM's Education Committee formed the working group to bring this report together. At the Town Hall meeting in Montreal, the general feeling from both academia and industrial representatives was that undergraduate CSE programs were probably best at the level of a minor (or equivalent local terminology) since a student needs to gain a thorough grounding in her or his primary discipline – which could be an applications field or might be mathematics or computer science.

There are good examples of major programs in CSE, perhaps most notably SUNY Brockport's adoption of not just an undergraduate major but its formation of a Department of Computational Science. As far as we know, this remains the only example of such an undergraduate department in the United States. Local politics alone may well be sufficient to make the adoption of a minor easier to achieve in most colleges.

In [1], Computational Science and Engineering is defined as *a broad multidisciplinary area that encompasses applications in science/engineering, applied mathematics, numerical analysis, and computer science. Computer models and computer simulations have become an important part of the research repertoire, supplementing (and in some cases replacing) experimentation. Going from application area to computational results requires domain expertise, mathematical*

modeling, numerical analysis, algorithm development, software implementation, program execution, analysis, validation and visualization of results. CSE involves all of this. Although it includes elements from computer science, applied mathematics, engineering and science, CSE focuses on the integration of knowledge and methodologies from all of these disciplines, and as such is a subject which is (in some sense) distinct from any of them.

The graphical representation of CSE in Figure 1 is one of several variations on this theme. We chose this one as illustrative of our belief that CSE is larger than the pure intersection of the three component pieces, but is nonetheless included in their union. That is to say that we believe CSE provides, and strengthens, the bridges connecting those components but should not become a separate "island". If CSE is a separate discipline, then it fails to serve its unifying and enabling role, rather it becomes self-serving.

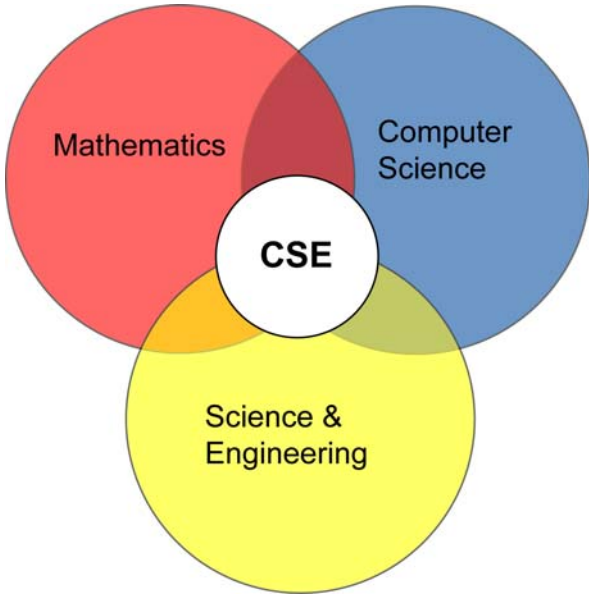

Figure 1: CSE includes, *but is greater than*, the intersection of mathematics, computer science and science & engineering.

One point we would like to emphasize in this document is that CSE is a legitimate and important academic enterprise, even if it has yet to be formally recognized as such at some institutions. Although it includes elements from computer science, applied mathematics, engineering and science, CSE focuses on the integration of knowledge for the development of problem-solving methodologies and robust tools which will be the building blocks for solutions to scientific and engineering problems of ever-increasing complexity. It differs from mathematics or computer science in that analysis and methodologies are directed specifically at the solution of problem classes from science and engineering, and will generally require a detailed knowledge or substantial collaboration from those

disciplines. The computing and mathematical techniques used may be more domain specific, and the computer science and mathematics skills needed will be broader. It differs from engineering or science in its focus on the development and innovative use of computational tools for the solution of complex problems.

There are several good literature sources for CSE. These vary in their primary objectives and therefore in the nature of their content. Articles such as [14], [16] are primarily concerned with undergraduate educational programs and their content. The earlier SIAM report [1] provided both a summary of then existing graduate programs, and some guidelines to help others interested in establishing such. Much of its content is also relevant to the undergraduate CSE community, at least with respect to general considerations. Many articles have been written and published on teaching particular aspects of a computational science curriculum. These include, as examples, [3], [2], [13], [15] as well as many papers presented and published in proceedings of the ACM Special Interest Group on Computer Science Education (SIGCSE) [23] and the IEEE Computer Society sponsored Frontiers in Education (FIE) conferences [19]. The Education Section of *SIAM Review* is a useful source of further guidance. Foundations such as Krell [5], and Shodor [35] have been valuable contributors to the development of materials and their dissemination via their own web sites and those of other projects. These activities have also resulted in some of the recent undergraduate texts focusing on different aspects of the CSE curriculum, such as [6], [11]. These books are notable in their emphasis on the use of somewhat real-world applications as the motivating theme for development of the curriculum material. There is of course a large array of texts concentrating on the traditional numerical methods and numerical analysis topics with a range of different levels and emphases.

One particular page, `www.krellinst.org/learningcenter/articles.html`, on the Krell website is valuable as a source of links to a number of articles and presentations covering, inter alia, curricular materials on computational aspects of atmospheric science and chemistry. This site also includes links to several of the relevant Journals, Institutes, K-12 projects, Government Labs, and so forth. The web-based courses link from the same root [5] has a good range of projects and other course materials in a wide variety of applications areas.

Many CSE programs place significant importance on the use of projects. General publications such as *Computing in Science and Engineering* [18] and *Scientific Computing World* [21] provide a wealth of current research applications of CSE at work in the research environment. *SIAM News* [22] also carries topical stories of new computationally enhanced advances in many areas of science and engineering. As well as providing a resource for propagating an understanding of the importance and central nature of computation in scientific research, these magazines can be a useful source of potential projects for undergraduate programs – in scaled down versions of course. *Computing in Science and Engineering* also includes a section on educational issues.

One of the noticeable trends is the growing influence of biological science on applied mathematics, and computational science in particular. The recently published report *Math-Bio 2010* [12] is devoted to the growth of mathematical

(and especially computational) biology within educational programs.

The "Computing Curricula 2001 Final Report" from the Joint Task Force on Computing Curricula of the IEEE Computer Society and the Association for Computing Machinery states the following:

> With the broad range of applications of computing in today's society, computer scientists must be able to work effectively with people from other disciplines. To this end, the CC2001 Task Force recommends that all computer science students should: Engage in an in-depth study of some subject that uses computing in a substantive way.

The report continues: "For many students, study of computing together with an application area will be extremely useful." Moreover, with one of fourteen focus groups concentrating on computational science (CN), the Task Force includes CN as part of computer science's body of knowledge.

Simulation Based Engineering Sciences (SBES) is defined as the discipline that provides the scientific and mathematical basis for the simulation of engineered systems (i.e., microelectronic devices, automobiles, aircraft, oilfields), (http://www.ices.utexas.edu/events/SBES_Final_Report.pdf). SBES fuses the knowledge and techniques of traditional engineering fields with the knowledge and techniques of fields like computer science, mathematics and the physical and social sciences. The SBES and the President's Information Technology Advisory Committee, PITAC, reports (http://www.nitrd.gov/pitac/reports), recommend that universities must significantly change their organizational structure and promote and reward collaborative research, and must implement new multidisciplinary programs that provide rigorous multifaceted education for the growing ranks of computational scientists. There is an urgent call for the nation to maintain leadership in scientific discovery. The integration of SBES into the educational system will broaden the undergraduate curriculum, thus giving students access to educational materials that demonstrate theories and practices that complement the traditional experiment/theory paradigm. In addition SBES will provide a rich environment for undergraduate research with students working in teams.

Many important issues concerning CSE education are still not fully resolved. A particular issue that is frequently posed is whether CSE should be housed in a separate department, or as an interdisciplinary program. While the majority of existing responses have favored the interdisciplinary approach, there are notable examples of successful programs in their own departments.

Other persistent, and related, questions are the issue of whether CSE represents an academic discipline in its own right, and the curricular content debate over computational science vs. scientific computing or computer science. It is *not* the intention of this report to attempt a definitive answer to any of these. The overriding belief that has been expressed repeatedly at the various open discussion events at SIAM meetings is that the right answer to any of these questions is necessarily local, depending on institutional politics, the spread

of disciplines represented by interested faculty, overall resource considerations, balance of the student body, etc.

The development of computation as the third leg of scientific research, led naturally to a growth in graduate CSE programs that has been followed by rapid expansion at the undergraduate level. In November 2003 the Krell Institute published their report on *Computational Science Education* [14] which includes a catalog of those programs that had responded to the survey on their web page. One striking aspect of the various programs is the variation in content and scope.

Some content elements appear to be common in the emerging undergraduate CSE curriculum: scientific programming, numerical methods/scientific computing, linear algebra, differential equations, mathematical modeling, and statistics are common mathematics components; advanced programming, parallel and high performance computing, and scientific visualization are commonly added where the program has its home closer to computer science; simulation, optimization, computational fluid dynamics, image and signal processing are among the offerings from some of the applications areas. Generically titled courses such as computational physics, chemistry or biology are often found.

A number of immediate inferences may be drawn. We suspect that almost all undergraduate (applied) mathematics programs offer the mathematical fundamentals of a CSE program. The availability of the computer science and applications related courses will vary much more with the nature of the individual school and the emphases of their departments. Another critical aspect is the out-of-class experience that is included. Projects (of varying length) are common; internships and more extended multidisciplinary projects are less frequent but provide great benefits to the students where they exist. These aspects are discussed in detail in subsequent sections of this report.

All our readers are aware that it is not always easy to create innovative undergraduate programs in our colleges and universities. This need was nicely summarized in [13] as follows:

> With the dramatic changes in computing, the need for dynamic and flexible computational science curricula becomes ever more obvious. Computational science has emerged, at the intersection of computer science, applied mathematics, and science disciplines (see Fig. 1) as a third component of science, in addition to theoretical investigation and experimentation. Mastery of computational science tools, such as 3D visualization and computer simulation, efficient handling of large data sets, ability to access a variety of distributed resources and collaborate with other experts over the Internet, etc. are now expected of university graduates, not necessarily computer science majors. However, the existing infrastructure on university campuses (human, technological, administrative) is often inadequate for the task. We described in [(their) 3] at least ten obstacles to a wider acceptance of computational science in undergraduate education; most of them are not technology related.

In Section 2, we examine some of the different models for CSE undergraduate programs and present some case studies drawn from existing programs. These programs include full undergraduate majors in CSE or in a particular computational science. In several cases undergraduate CSE programs entail minors, (sometimes called *emphases* or *concentrations*) which can be taken along with a major in mathematics, computer science, or a scientific or engineering field. Commonly, decisions on the nature of a particular program are founded on pragmatic considerations: what can we get approved at our university. In other instances programs consist of special individual courses that are usually project-based, and are distinct from the traditional computational mathematics classes, such as numerical methods. As will be seen, there are many common features and several more individual aspects. The use of multidisciplinary "research" projects and encouragement (or requirement) of a computational science internship are often cited as existing or aspects to be desired. Topical content also varies and raises interesting questions. Is a conventional programming course essential? Should students be exposed to professional software packages – and when? When is high performance computing introduced/discussed? Is statistics a core component? How much mathematical background is required? What proportion of the program is devoted to applications fields? The sample programs presented illustrate a range of different answers.

What should the content of a CSE program be? Very often several of the components that will be needed will already be in place. Indeed if this is not the case, it may be difficult to develop a good program in your school. The key components are that courses, or other aspects of the educational experience should provide core competencies in most of the following topics: Simulation and modeling; Team-based projects; Effective technical analysis and presentation; Programming (in a high-level language) and algorithms; Applied mathematics; Numerical methods; Parallel programming; Scientific visualization; and a Research or Professional Experience

It is of course unlikely that any one program will provide a thorough and deep understanding of all these topics, but the list provides a useful benchmark for designing an undergraduate CSE program. The discussion in Section 2 provides more detail.

In Section 3, we highlight the valuable role that internship programs can play; and in Section 4, we attempt to delineate the *Needs that undergraduate CSE should address.* There are no universal, and realistically achievable, answers to questions such as: What skills are required for jobs in industry following the undergraduate degree? Or: What skills are required for successful transitions into graduate CSE education? These questions are discussed, and some recommendations for the essential attributes of CSE graduates are developed from the discussion. CSE careers form the central theme of Section 5. Some of the possibilities are illustrated through profiles of recent graduates of CSE programs, from employers of such graduates, and from graduate programs they enter.

One necessary follow-up to this report is that SIAM should continue its leadership in the field by conducting an extensive survey of existing programs

for inclusion on the web. It is hoped that the increased awareness created by this report will spur a high level of response so that everyone can gain from the collective experience of all practitioners. Section 6 contains our conclusions and recommendations.

2 Nature of CSE Undergraduate Education

2.1 Introduction

The undergraduate educational system has been slowly but steadily reacting to the necessity of educating undergraduate students in CSE by creating a variety of curricula that best fit the different types of institutions.

We believe that the undergraduate arena is the most important segment of the educational pipeline, since it prepares the science/math teachers for the High School environment, invigorates students to pursue graduate studies in cutting edge technical fields, and produces a vast number of future employees for industry and the "knowledge based" economy. Therefore, it is critical that computational science curricula/programs are a viable option for every undergraduate STEM major to provide students with skills highly desirable for future employment or graduate work.

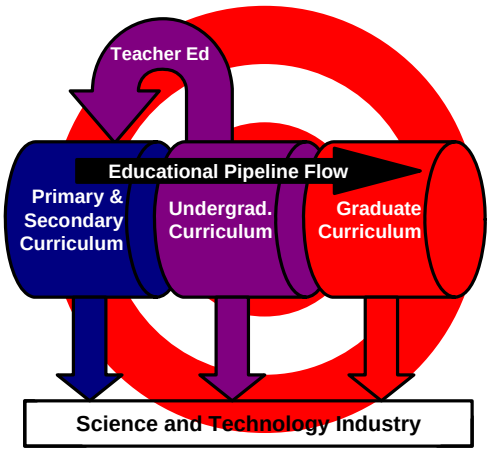

Figure 2 The CSE Educational Pipeline

Figure 2 shows the central position of undergraduate CSE education in the pipeline. It is the one place that feeds three different markets. The primary objectives of preparing graduates for graduate studies in CSE and for careers in industry are joined by a potentially critical contribution: preparation of good

teachers for the K-12 system who have a thorough appreciation of the integrated nature of the STEM disciplines and the use of relevant applications and technology in problem-solving for mathematics and science education. There is increasing activity in K-12 outreach in the CSE arena.

A number of motivational factors is responsible for the development of a (flavor of) CSE program at a specific campus. Those may include:

- Future jobs of a technical nature are and will be requiring new skills directly related to computational science. For graduates whose primary field of employment is one of the STEM fields, the most important skills are scientific and mathematical, and the ability to synthesize information. All of these skills are essential components in computational science education.

- Observations that the typical computer science graduate does not have the appropriate level of modeling, mathematics and science background needed for future technical employment. Similarly, a typical science/mathematics major does not posses adequate background in computation.

- CSE, as a research field, facilitates the integration of mathematics, computing and science/engineering and holds great promise for solving many challenging problems facing humanity. A similar integrative approach must be part of the undergraduate STEM experience.

- Most STEM fields are becoming more computational. Much of science and engineering is now performed *in silico*. Computational simulation and experimentation with "what –if" scenarios not only allows the developers of a product to get more creative, but also saves time and money.

- Numerous reports and articles are stating the importance of computational science. For example, the PITAC report [9] indicates that computational science is one of the most important technical fields of the 21st century because it is essential to advances throughout society. Publications such as [18], [20], [21] carry many examples of CSE in practice. These can be useful sources of motivational material for classes.

- There are growing opportunities for funding the development of computational science curricula and educational materials focused on the undergraduate arena.

There are several good literature sources for computational science education. Articles such as [14], [16] address some of the existing undergraduate computational science programs of study. As stated above, the SIAM report [1] provided both a summary of then existing graduate programs and some guidelines to help others interested in establishing such. The main objectives of this section are: i) to provide a quick overview of the different "flavors" of undergraduate computational science educational programs; ii) to indicate important common features and characteristics among the various "flavors"; iii) to present

a short survey of existing curricula materials and their common threads, learning outcomes and student competencies; and iv) to present a collection of case studies – descriptions of computational science programs for various types of institutions.

Yasar and Landau [16] provide an excellent characterization of the evolving nature and stages of computational science education. Specifically, they indicate: i) the recognition-conception stage (1980-1990) recognizing the growing importance of computation (alongside theory and experimentation) in practising and advancing science and engineering; ii) the infancy stage (1990 – 2000) where computational science courses started at the graduate level; iii) the early growth stage (2000 – 2010) where both graduate and undergraduate curricula are being developed at a rapid pace; and iv) the adult stage (2010-beyond) where the authors predict that the number of computational science programs will continue to grow in a more rapid pace. In addition to growth of computational science program, we predict an even more rapid growth to specialized Computational X (X = STEM discipline or Finance) courses. It has been stated that pressure for the development of such courses will be attributed to the needs of industry/national labs, the desire of STEM related departments to modernize their curriculum, the desire of students to apply their general computational modeling skills, the quest for external funding, and the passion of dedicated faculty who believe in the power and promise of computational modeling as applied in STEM disciplines.

2.2 Models for CSE Programs

The official designation of a computational science (CSE) program may be different across institutions, but the overall structure is very similar. The most common flavors of undergraduate computational science programs in the U.S. include:

- B.S. degree in CSE
- Comprehensive Minor program in CSE
- Minor program in CSE
- Emphasis or Concentration in CSE
- B.S. degree in Computational X (X = STEM discipline or Finance)

Even though there are no official curriculum guidelines or nationally accredited CSE programs, most of the above flavors exhibit a number of common features.

One way to characterize the similarities of the programs is to examine the number and type of courses. In fact, almost all of the above "flavors" of undergraduate CSE programs include a common "core" or collection of courses in: Calculus (2 course sequence); Programming (at least one course); Introduction to Modeling-Simulation (or Computational Modeling); Numerical Analysis (or

Scientific Computing); and either a course in Visualization or a more advanced course in Computational Modeling. Another common feature of most CSE programs is the requirement for an internship in the form of an undergraduate research experience. This is currently being implemented as a separate credit bearing course, a non-credit internship, or as part of an upper division computational modeling course. All existing CSE programs recognize the importance of independent research by undergraduates since such experience provides the opportunity for a student to work (over a semester, summer or for a longer period of time) on a real-world problem with either a faculty or an industry mentor. These research experiences are being implemented as single or team-based projects and include some form of written or oral presentation on campus or at a professional conference. The next section provides more information and examples for such valuable undergraduate CSE experiences.

Another way to view the common features of CSE programs is to examine either the "Learning Outcomes" or the minimum set of "Competencies" that an undergraduate student must achieve through a set of core courses.

Yasar and Landau [16] provide a set of "Student Learning Outcomes" for an undergraduate CSE education. Indeed, these are very common among the programs listed in the survey by the Krell Institute [14]. The common learning outcomes include:

- Learning of a high-level language
- Acquiring knowledge of applied mathematics
- Demonstrating knowledge of computational methods
- Learning the basics of simulation and modeling
- Applying effectively computational modeling techniques to an application area from STEM disciplines
- Interpreting and analyzing data visually
- Applying computational-modeling skills
- Learning to communicate the solution process effectively

A similar approach is being implemented for the development of CSE statewide programs based on set of common competencies by the newly formed Ralph Regula School of Computational Science (http://www.osc.edu/education/regula), a statewide (Ohio based) virtual school focused on the emerging and diverse area of CSE. Currently the school is directed by the Ohio Supercomputer Center-OSC (http://www.osc.edu) under the auspices of the Ohio Board of Regents. Its long-term education mission is to successfully infuse CSE in all segments of the educational pipeline (K-20), including the creation and offerings of certificate programs for adult learners. The current priority of the virtual school (under the direction of Dr. S. Gordon) is the development of specific competencies and standards for a statewide CSE curriculum at the undergraduate level.

The agreed upon high-level competencies by a set of faculty and members of an industry advisory committee include competencies on the following areas:

- Simulation and modeling (conceptual models, accuracy, use of modeling tools, assessment of computational models, team-based projects, effective technical analysis and presentation)
- Programming and algorithms (a high level language, elementary data structures and analysis)
- Applied mathematics (concepts in a calculus sequence as well as differential equations and discrete dynamical systems)
- Numerical methods (errors, non-linear equations, solving systems of linear equations, interpolation—curve fitting, optimization, Monte Carlo, ODEs and PDEs)
- Parallel programming (knowledge of MPI and OpenMP)
- Scientific visualization (basics including grid representations, pipeline, rendering, and vector visualization)
- Research experience (independent research, presentation of solution methodologies)

Over the last 5-7 years, we have experienced a rapid growth in the availability of undergraduate CSE educational materials primarily as a product of funded grants (i.e., CCLI and Cyeberinfrastructure programs at the National Science Foundation; W.M. Keck Foundation). For the most part, the developed educational materials are:

- Web based
- Self standing; comprehensive modules
- Guided by the problem-based approach centered on applications from STEM disciplines
- Designed for using the inquiry based pedagogy (i.e., promoting the exploration of the "what –if" scenarios)

A collection of freely available CSE materials developed through grants from the National Science Foundation (CCLI program) and the W.M. Keck Foundation (Keck Undergraduate Computational Science Education Consortium) can be located at: http://www.capital.edu/keck-consortium and http://www.capital.edu/futureofscience). The educational materials adhere to a common paradigm: Problem — Model — Method — Implementation — Assessment. Such a paradigm has been used extensively in the teaching of CSE courses at the undergraduate level for the past decade, and was proposed by the Undergraduate Computational Engineering and Science Group (http://www.krellinst.org/UCES) in the early 90s.

An extensive and diverse collection of a variety of CSE resources that goes beyond the STEM disciplines is being assembled at the Computational Science Reference Desk—CSERD (http://www.shodor.org/refdesk/). The CSERD, a Pathways project of the National Science Digital Library (http://nsdl.org/) funded by the National Science Foundation, aims to help students learn about CSE and to help teachers incorporate it into the classroom. CSERD attempts to: i) Collect a catalog of quality resources from across the internet; ii) Provide a forum for the Verification, Validation, and Accreditation of catalog items both by users and by expert reviewers; and iii) Create original CSE resources for use in education.

It should be noted that the successful development of a specific flavor of a CSE program depends on the structure and mission of a particular university, the collection of faculty expertise and most importantly on pragmatic considerations (i.e., which and how many courses can be approved by the institution? What are the local politics?). The authors of this report believe that the establishment of a minor program in CSE (or any of its variations) is much more pervasive and easier to achieve than a B.S. degree curriculum in CSE (or Computational X). Some reasons to support this view include: i) CSE is a multidisciplinary area and a minor program in CSE complements any traditional STEM major (the latter provides the necessary disciplinary depth); ii) a minor program is not viewed as a threat for a "competing major" among well established traditional majors; iii) a CSE minor that contains an array of Computational X courses can serve as a common arena for true multi-disciplinary collaborations of faculty and students that belong to different STEM based departments; it can also serve as a catalyst for reducing (or even eliminating) existing compartmentalization among departments.

Yet another model has begun to emerge as an alternative to the "Discipline Major – CSE Minor" model. As a result of the observed need for developers of CSE solutions to have a deeper understanding of the underlying mathematics and computer science, there is support for a "Computational Applied Mathematics major – Applications field minor". The major part of this program would not be the traditional mathematics major, though it would certainly include significant pieces of it. It would have a strong emphasis on applied mathematics with a larger than usual computational component. These fields offer good opportunities for project-based learning and team work as well as exposure to the relevance of mathematics to real world problems. This major entails a greater exposure to computer science including high-level language programming, data structures and algorithms, and scientific visualization as outlined for the general CSE content above. The applications field can be in any STEM discipline or a more general engineering science. Again, whether it is called a minor, a second discipline, a concentration, or an emphasis will vary according to local terminology.

At this point we are unaware of any programs of this nature that are presented in this way. However an appropriate set of learning experiences can probably be acquired through appropriate choices of elective classes in an applied mathematics major with a good computer science core and a minor in some

applications area. Proper development and advertisement of such packages is needed in order to attract the appropriate students and ensure they make good choices.

2.3 A Few Examples

The following give a short overview of the various flavors of CSE programs at the undergraduate level. The intention is to provide the reader with samples of existing "flavors" of successful CSE programs. Clearly there are many other successful examples of CSE programs [14], [7].

B.S with a major in Computational Science, SUNY College at Brockport

http://cps.brockport.edu

Students take courses in computational science (computational tools, computational modeling and simulation), applied-computational mathematics and support courses in a variety of STEM disciplines. The program also includes courses in scientific visualization and high performance computing, along with an array of electives in Computational X fields (X =Physics, Fluid Dynamics, Chemistry, Biology, Finance). Undergraduate research experience is required.

Comprehensive Minor Program in Computational Science, Capital University

www.capital.edu/futureofscience

The minor program was established in 2004 and has been supported by a number of grants (i.e., National Science Foundation, W.M. Keck Foundation). Its 22-credit curriculum was developed by the collaboration of all math- and science-based departments within the college of A&S along with the finance and economics departments within the School of Business. The minor program supplements a number of STEM majors and provides unique skills to students that complete the program. The curriculum contains a set of core courses (Computational Science, Programming, Differential Equations/Dynamical Systems, Numerical Methods); and a set of electives (Computational X, Parallel and High Performance Computing, and Scientific Visualization). Specialized courses include Computational: Biology, Chemistry, Environmental Science, Physics, Psychology, Finance and Economics. Undergraduate research experience is required.

Minor Program in Computational Science, University of Wisconsin – Eau Claire

www.cpsc.uwec.edu

The interdisciplinary curriculum at this liberal arts college was developed with the collaboration of the Biology, Chemistry, Computer Science, Mathematics, Geography, and Physics/Astronomy departments. It consists of a calculus sequence, a two course sequence in computational modeling, a course in mathematical modeling and a course in numerical methods. A computational science practicum is required.

Minor in Computational Science, Clarkson University
www.clarkson.edu/mcs/math/undergrad/minors.html#computational

This minor serves as an example of bringing together existing courses in applied mathematics, computing and applications fields to create a minor in a largely science and engineering based institution. The minor itself lacks an internship or professional experience. This is because the newly instigated core curriculum at Clarkson requires such a professional experience as part of the graduation requirements for every degree program. Students in the CSE minor will therefore gain that experience through their major. The title also reflects local politics with "and Engineering" being rejected on the grounds that no engineering classes were *required* for the minor.

Emphasis in Computational Science, Wofford College
www.wofford.edu/ecs

At Wofford college the Emphasis in Computational Science is a truly interdisciplinary program among science, computer science and mathematics. Students major in one of the math or science disciplines and complete a required summer internship in a CSE sub-field. The five required courses for this "flavor" of CSE program include: Programming, Data structures, Calculus I, Modeling and Simulation and a course in Data and Visualization.

B.S in Computational Physics, Oregon State University
http://www.physics.orst.edu/~rubin/CPUG/

The Oregon State Board of Higher Education approved the Computational Physics degree in October 2001. The specialized B.S. degree includes five courses with computational modeling ingredients: a two course sequence in scientific computing, a course in computational physics simulation, a course labeled advanced computational physics laboratory and a computational physics seminar. The degree program offers a well balanced blend of physics, computational modeling, computer programming and applied mathematics courses.

B.Sc. in Computational Engineering, Universitaet Erlangen-Nuernberg
http://www9.informatik.uni-erlangen.de/CE/06Bachelor

The Erlangen Computational Engineering program includes a 3-year Bachelor degree and Master and PhD degrees. The objective of the CE program is to give students a genuine interdisciplinary education from the ground up. It consists of roughly equal number of credits in mathematics, computer science, and an application field that is currently limited to one of the engineering disciplines (Micro Electronics, Automatic Control, Thermo- and Fluid Mechanics, Sensor Technology, Material Sciences, Applied Chemical Engineering). Students are required to take a selection of core courses that include the traditional four semester engineering math sequence and the most fundamental courses from the computer science curriculum.

3 The Value of Internships

Internships can be extremely valuable for a student in any area, but particularly in CSE, where interdisciplinary teams work on large, "real-world" problems. Internship experiences expose students to a wealth of new ideas, techniques, and applications that enhance their knowledge of CSE and make classroom education more meaningful. As an advantage to the host institution, undergraduate interns can make significant contributions to its research. Moreover, a student can leverage an internship to obtain subsequent professional experience, including admission to a better graduate school program, a graduate assistantship or fellowship, or a better professional position than would have been possible otherwise. As an added benefit, most internship positions allow students to visit different areas of the country and to meet students from other parts of the country or world. For these reasons, most undergraduate CSE programs encourage students to obtain summer internships or research experiences, and a few, such as Wofford College's Emphasis in Computational Science (ECS), require an internship.

Figure 3 Liz (biology and computational science) with her mentor, right, from Oak Ridge National Laboratory (ORNL). ORNL sponsored Liz to attend the Supercomputing Conference, where she made a poster presentation. She continued her research through the school year and the next summer

Considering the importance of this component of a CSE program and the inexperience of students in pursuing positions, faculty members often must actively help in finding opportunities, developing research proposals, and making professional presentations following internships. Advisors must encourage the CSE student to start the process early in the fall, because many deadlines are

early and the application process can be time consuming. Because positions are very competitive, the student usually should apply for a number of internships.

> Amanda, a Biology major and ECS student at Wofford College, provides an example of a "go-getter" who used internship experiences to learn and to create opportunities. After her sophomore year, she obtained an internship at the Greenwood Genetic Center through a faculty contact. During the summer, she developed a Perl program to automate the linkage analysis that the laboratory performs on disease genes, created a database on the diets of Honduran women, and used SAS (Statistical Analysis System) to examine the relationship between diet and birth defects. She presented her work at the SIAM Annual Conference's first Minisymposium on Undergraduate Research. With contacts and recommendations from the summer, Amanda obtained another internship at Massachusetts General Hospital/Harvard during January. With multiple successful research positions, she obtained yet another at National Institutes of Health (NIH) after her junior year; and by the end of the summer, NIH had offered her a year long IRTA Research Fellowship at NIH's National Institute of Child Health and Human Development. Amanda is currently pursuing Ph.D. in genetics, a very computationally oriented field, at the University of North Carolina and is working on a project that blends many of the topics researched during her internships.

Before applying, a student should have a good resume that explicitly and prominently states his or her computer science, mathematics, and science background, including coursework. A school's career services office and career-oriented websites can assist in writing a resume and in finding opportunities. A faculty member in the area of CSE should review the resume carefully to make sure it contains all the information that potential mentors want to know. At that time, the advisor should caution the student to write professional, grammatically correct, respectful, and succinct emails to prospective mentors.

Several sources on the World Wide Web have links to a number of organizations with internships, and most large companies and government laboratories have established internship programs. Also, each year, the National Science Foundation funds a number of Research Experiences for Undergraduates (REU's) at universities around the country.

Often, some kind of personal contact is very beneficial in securing a position. In high-tech regions, the student should ask friends, family members, and faculty if they are aware of any opportunities or if they know someone who might help. More commonly, faculty members can obtain contacts by talking to people at conferences and workshops, asking colleagues at other institutions for suggestions, or employing their own faculty-lab/company connections. A direct contact at an organization is far more effective than dealing entirely with the human resources (HR) department. HR might not understand who in the

organization can use someone with CSE background. Also, direct contact with a scientist might result in the creation of an internship position just for the student.

Faculty members' recommendations often play a major role in selecting interns. It is important that letters mention specific accomplishments of a student, especially if they are relevant to a position.

Unfortunately, some created or even established positions, particularly if far from the home institution, do not provide adequate funding for the student. Thus, it is very helpful if the school can provide supplementary money for worthwhile projects. If a school can subsidize internships, then students will have more opportunities. Students can seldom afford to lose money over a summer but are usually willing to "break even" for a rewarding experience.

Figure 4 Paul (REU at Virginia Commonwealth University), Christine (Honeywell Aerospace), and Ricaye (ORNL) present posters on their computational science internships.

Two professors at Wofford College visited the Molecular Graphics Laboratory at The Scripps Research Institute to make initial inquiries for students. The following year, one of these professors emailed a contact at the lab inquiring about a possible internship for a computational biology student and attaching his resume. Coincidentally, that scientist needed a student who knew biology, chemistry, computer science, mathematics, and physics to program a printer to produce "tangible models", or three-dimensional models of molecules, such as hemoglobin. Realizing that the recommended student, Wesley, had the rare interdisciplinary knowledge needed, the scientist requested and obtained an REU as an extension to his existing NSF grant. Because of the expense of travel to and lodging in California, the school provided additional funding to Wesley.

Paul's internship at Virginia Commonwealth University's NSF REU Bioinformatics and Bioengineering Summer Institute illustrates the organization of much scientific investigation and emphasizes the importance of training science students to become active participants in computation as well as theory and experimentation. At the laboratory, "wet lab" and "dry lab" teams work closely to investigate the life cycle of Trypanosoma cruzi (T. cruzi) parasite, which causes Chagas' disease affecting 14 million people. With data and questions generated by experiments from the wet lab team, the dry lab team is developing a high performance computational model of T. cruzi's lifecycle. The resulting simulations are helping scientists to understand the parasite and to develop effective strategies for combating the menace. With an education in biology with computational science, Paul was able to work and communicate effectively with both teams. Such internships can help students evolve into scientists who can work effectively in teams using theory, experimentation, and computation to investigate important scientific questions. The bridge-building that is inherent in CSE was therefore evident even at this early stage in the student's career. He was able to provide the links that led to real scientific advances in the project.

Besides the intricacies of obtaining internships, most students need some instruction on professional etiquette and insight into the anticipated experiences. They should be cautioned to accept or reject an internship offer promptly. For example, one student with two offers took so long to respond that both positions evaporated. It is natural to be fearful of a new situation, especially one involving a project for which the student knows little or nothing. Faculty members can reassure prospective interns that almost every other intern shares their anxiety and anticipation. A mentor usually does not expect a student to be familiar with the particular project and will help the intern gain the necessary background. A great benefit of an internship is having the experience of working with a team on something beyond the classroom. Thus, before the experience begins, an undergraduate advisor can encourage the student to seek help from team members as needed.

Most teams ask an intern to write an abstract of accomplishments during the summer and to make a final project presentation. The abstract and presentation can serve as a basis for a later conference paper. Keeping a daily journal is very helpful in summarizing the summer's activities and documenting its many achievements.

After the internship, the student's advisor should contact the intern's mentor to obtain an evaluation of the student's performance. Such an evaluation can provide valuable insights to note in future recommendations for the student. Moreover, the conversation can help to build rapport and increase the likelihood of other internships at the same organization.

In the fall, faculty members can have the interns give presentations on their work. Student discipline-related organizations, such as local student chapters of SIAM, the Association for Women in Mathematics, the Association for Computing Machinery, or Tri-Beta, are usually delighted to have presentations. Students and faculty can learn from the presentations, which might also inspire other students to pursue CSE or seek internships. Moreover, such experiences add to a student's poise in such situations and can help to prepare him or her for conference presentations.

Most conferences have sessions for undergraduate papers or posters. For example, SIAM's Computational Science and Engineering Conference has a series of minisymposia on undergraduate CSE research. A few conferences and organizations, such as SIAM, provide assistance to some students making presentations; and universities and colleges also might pay all or part of students' expenses. However, students usually not are aware of conferences and are often apprehensive about making presentations. Thus, professors must often encourage former interns to participate in appropriate venues. Many conferences only require a student to submit an abstract of one or two paragraphs instead of a lengthy paper. A professor can help further by reviewing the abstract or paper before submission. Through a conference research presentation, a student can gain confidence, make valuable contacts, generate a significant addition to a resume, and continue learning how to be an active professional computational scientist.

Figure 5 Internships each summer, including ones at Lawrence Livermore National Laboratory and Los Alamos National Laboratory (LANL), prepared Diana to accept a position at LANL after competing her master's degree.

There is so much that can be gained from internships. Amanda, who was the first intern example above, summarized their benefits well, as she wrote, "internships are not only important, but necessary to gain advancement in your field of interest; they lay the foundation for your future career. My four internships provided me with experiences I could not have gained in school. Besides giving me hands-on experience in various areas, these internships taught me professional responsibility, allowed me to grow personally, and created opportunities for my future. While working at some of the best facilities in the nation, I meet and had opportunities to work with many brilliant people, who gave me their honest advice and opinions on their subject and profession. From these internships, I was able to determine the career path I wanted to take."

Frank, who as an undergraduate had majors in physics and mathematics and took computer science classes through data structures, also discovered his career path of computational astrophysics through two internships at the Jet Propulsion Laboratory (JPL). During one summer at JPL, he worked with the Near Earth Asteroid Tracking (NEAT) team and developed software (HAVANA) to perform mathematical computations and to access quickly images of specific moving objects from JPL's extensive observational archive. Towards the end of the summer, NEAT observed a previously unknown asteroid for twelve days but needed earlier photographs to determine the object's exact path through space. Using HAVANA, they found the asteroid, previously undetected, captured on film from 1993. This archival discovery made it possible to calculate the asteroid's orbit with minimal error. Delighted, Frank's mentor at JPL recognized the contribution of this undergraduate intern by naming an asteroid in his honor. Graduating in 2005 from Washington University in St. Louis with a Ph.D. in Earth and Planetary Sciences, Frank currently holds a post-doctoral research position at the Johns Hopkins University Applied Physics Laboratory working with the Compact Reconnaissance Imaging Spectrometer for Mars onboard the Mars Reconnaissance Orbiter.

Eric is a 2005 graduate from Capital University. Eric's educational background involved a B.S. degree in Computer Science along with minors in mathematics, religion, and *computational science,* and a number of courses in biology. His educational experience involved a number of research endeavors which were possible due to his multiple dimensional background and his training in computational science. Eric's ability to build bridges among traditional disciplines set him apart from many other peers.

During his sophomore year, Eric participated in a research project under a former consultant for Battelle and president of Innovative Thinking Inc. Eric and a fellow student built a mathematical model to describe smoking dynamics. The model followed hourly nicotine fluctuations in the user's blood and how that contributed to their daily smoking habits and has been implemented in STELLA. The integration of multiple disciplines made a unique project to present at the National Conference for Undergraduate Research (April 2004). In addition, the modeling project has been given serious consideration by the Tobacco Use and Prevention Foundation in Ohio.

Eric is currently the lead developer for the Genome Research Institute Discovery Platform (GRIDP). Typically a researcher in Biology/Biochemistry must understand command codes to use high performance computing (HPC) programs and systems, often used in bioinformatics, drug discovery, and other computational fields. Although much research could benefit many biologists and chemists do not themselves have facility with such computational methods. GRIDP solves this problem by creating graphical user interfaces for these pro-

grams and by bridging the gap between sciences and computational science, thus enabling researchers and students to use HPC systems across the state of Ohio for computational science research and learning. GRIDP is serving a state-wide drug discovery platform, with the potential of becoming a state-wide research platform for all computational disciplines (http://www.osc.edu/press/releases/2006/gri.shtml).

Cross discipline communication and collaboration is the medium from which new endeavors can be created. GRIDP began as a lunch conversation between Eric and a biologist and computational scientist, about the difficulty of using many computational programs. This insight would never have resulted from a discussion between two computer scientists, or two mathematicians, or two biologists. Eric's undergraduate education in computational science at Capital University, gave him a unique entrance into growing industries of bioinformatics, drug discovery, and other fields of computational science.

4 Needs that Undergraduate CSE Education Must Address

In many ways the answer to the question implied by this section title has been answered already at least in terms of basic content. What is less clearly stated is the overall picture of what an undergraduate CSE student should have achieved at the completion of his/her BS degree.

Expressed in broad terms, the overall needs are a combination of disciplinary skills and cross-disciplinary skills, learning how to learn, ability to work in a team, adaptability, perseverance and an interest in solving problems that may be multi-faceted. Acquiring these abilities is, of course, challenging. Realistically all we can expect is the beginning of the process. Like most mathematics, science, and engineering undergraduate programs, a CSE program should be seen more as a starting point than a finishing one.

By the nature of CSE (look again at Figure 1), the successful undergraduate CSE student will have skills in applied mathematics, computing including some parallel or high performance computing, and at least one application field. Typically, the students will have had the experience of working on extended projects, preferably in a team-work setting. These projects will have necessitated the integration of the various skills and therefore should themselves be interdisciplinary.

Sometimes this integration can be achieved through a team-taught projects course. In other instances integration may be via a senior project or thesis similar to engineering students' senior design projects. Achieving this objective through professional experience gained in an internship or co-op is perhaps the ideal.

As with many undergraduate programs, there is a conflict between providing the necessary academic depth of background for potential graduate studies and preparing students for the workplace. In some senses, this difficulty is exacerbated, and in some senses obviated by a study of CSE. The exacerbation

derives from the breadth of the multidiscipline which in itself makes fitting all the material into a curriculum to prepare students for graduate work in perhaps mathematics, perhaps computer science, or a chosen application field is nigh impossible. The obviation on the other hand stems from the students' abilities to adapt their prior knowledge to solving new problems and their ability to synthesize different aspects of their background.

Figure 2 includes a third outflow from the CSE educational pipeline: feeding CSE-trained students into the K-12 system. Pre-college teaching is an important area into which the right students should be encouraged to move. While this may not have significant impact on the course content, it does provide additional opportunites for professional experience through educational outreach programs. Many states now have programs that can be used to support such activities. These often allow undergraduate students to work in K-12 classrooms or extra-curricular activities such as coaching local MATHCOUNTS or First Robotics teams. Professional experiences of this type can be very valuable in improving students' understanding and their communication skills in particular.

Figure 6: Student Coaches helped local schools prepare for MATHCOUNTS

A ideal CSE program would achieve all of these objectives – and probably several more besides.

The desirable curricular content and learning outcomes of undergraduate CSE programs have already been addressed in the introduction to Section 2. It is worth repeating some of that "shopping list" here to re-emphasize the key features. The content knowledge should typically include at least: Calculus (2 course sequence); Programming (at least one course); Introduction to Modeling-Simulation (or Computational Modeling); Numerical Analysis (or Scientific Computing); and either a course in Visualization or a more advanced course in Computational Modeling. Of course, students whose major is in CSE or in any of the constituent disciplines will have significantly greater background in their particular area of specialization.

Although the nature of the undergraduate research or independent project work will vary from program to program, there is near unanimity that such

experience should be included. In some sense this is merely confirmation of a growing trend in almost all undergraduate educational programs. The virtues of teamwork, project-based learning, computation and communication skills are recognized almost universally. What may set CSE programs apart from others is the extent to which this philosophy has been not merely followed but actively fostered, assuming a critical and central role in the programs.

What skills are required for jobs in industry following the undergraduate degree? This issue emerged as a central theme of the discussions at the Town Hall meeting at the SIAM Annual Meeting in Montreal, 2003. The general feeling was that the specific knowledge content was less important than the ability to think and work through problems. Industrial employers of CSE or other technical graduates recognized that there will be extensive on-the-job training in the specifics of their industry.

From an industry perspective, a background especially at the undergraduate level in CSE is becoming ever more important. As already pointed out simulation based engineering science is essential to industrial competitiveness. It helps to shorten time to market for product development. It allows for quick prototyping and designing of new products. It helps in testing of products through simulations. As industry relies more on simulation based engineering sciences, a greater understanding of this discipline is essential. Many people start their industry careers with only a bachelor's degree in some scientific discipline. As they rise through the company gaining experience they become managers. Having people with a fundamental understanding of computational science, mathematics and engineering as opposed to picking it up along the way will help businesses make good decisions when it comes to applying this discipline to their business. At this time and for this reason, the more undergraduates produced with a basic understanding of CSE the better business will be positioned to use CSE.

To give some idea of the far reaching impact CSE is having in industry, we mention some industry segments that are currently using CSE in their business. CSE has been a factor in the aerospace, automotive, chemical, computer, electronics, petroleum and pharmaceutical industries for some time. Industries that might not come immediately to mind that are now using CSE include banking and finance, digital media especially content creation, consumer products, manufacturing and processing and even in transportation. With the impact that computation has had in the sequencing of the human genome, CSE is playing an increasing role in the life sciences and healthcare. The need for trained computational scientists at all levels in the healthcare and life sciences continues now and for the foreseeable future to outpace the supply.

When industry seeks to fill a position, they look for a person who has the expertise to do the immediate task and the versatility for future, as yet to be determined, assignments. For this reason, it is important that training in CSE emphasize a strong foundation in a traditional discipline. An individual with expertise in CSE will need to have good computational skills. Some of these skills need to be in traditional programming languages as many current computational science, mathematics and engineering problems often include the use of legacy codes that may need to be modified or extended. Further, CSE

skills need to include an understanding of computational issues such as error analysis, computational stability, and performance to name a few.

The added breath from exposure to a second discipline that an individual trained in CSE will pick up will be helpful in an industrial position. It will allow the individual to transition to new problem areas because they speak another discipline's "language" and are trained to pick up new technical "languages" easily. Time is money in industry. Quickly determining a solution, even a negative one, is important. Discovering that an approach will fail early can save a company significant investment. As the CSE trained individual experiences the application of computation to a discipline, he/she has the breadth to not only transition to a new area but can drive the change necessary to make a transition to happen.

For an industrial position, training in CSE provides a background that industry often seeks in an individual to hire. For instance, many CSE trained individuals will have good problem solving skills. This is important especially when the problem itself is ill defined. In addition, a CSE undergraduate will often be exposed to working as part of a team. In industry, individuals with a variety of backgrounds work together. The team experience for the CSE undergraduate provides the opportunity to learn how to communicate ideas and concepts quickly and persuasively. It also helps to see how one's work fits into a bigger effort and how to communicate the impact of one's individual effort to the larger project. In doing this, one often must adopt different perspectives and communicate the same information in different ways. For example, when communicating with company management, one may need to speak in terms of the financial impact of the project, while speaking with technical people within the company, the scientific impact may be conveyed, and finally for purposes of advancing in a new area, the innovative technology is described. The ability to communicate on different planes is an important aspect for success in an industrial career.

Many CSE programs try to incorporate internships into their programs. While this is laudable and can be of significant value to the students, companies, unlike the academia and national laboratories, do not have as part of their mission to teach and train people. Faculty often doesn't understand this and wonder why it is difficult to find industry internships for students. Companies hire students as interns or for the summer with very specific backgrounds to get a short term project done while paying a minimal amount and leveraging existing staff. Student interns must already have specific skills, must be self starters, quick learners, and not be a burden to over-extended personnel. While industry budgets and budget cycles may be an issue to inhibiting internships, even if funding for the interns comes from elsewhere, if the intern cannot make significant contribution in a short time on his/her own, the impact from the company's perspective will be negative. Those in companies that seek to support projects for interns must do a lot of work to mitigate the negatives while providing for a meaningful experience for the student intern. This is not easy.

One way faculty can help the industry internships situation, is to collaborate

with industry colleagues. A particular project may be spun off for a student who already has significant background or can be prepared by the faculty member and thus can make substantial progress in a short time.

5 CSE Careers

In many senses the opportunities and needs for students embarking on CSE careers can be summarized by the experiences described below by one recent graduate who graduated as a Chemical Engineering major *before the CSE minor at Clarkson was available.* Sam took more mathematics classes than were required for his major, but did not have any computational classes. The following story describes how he has adjusted and how his very first significant assignment was one which fits squarely in the CSE portion of Figure 1. There is little doubt that Sam would have added the CSE minor. For the rest of this section, Sam tells his story in his own words.

I am a recent graduate of Clarkson University in Potsdam, NY and am currently employed by The Procter & Gamble Company in Cincinnati, Ohio. I work as an engineer in R&D designing new Oral Care products after graduating with a bachelor's degree with great distinction in chemical engineering and a minor in mathematics from Clarkson University in 2003. I currently live in Cincinnati, Ohio after being raised in upstate New York. In my free time, I adopted and am working on rehabilitating a park behind the building in downtown Cincinnati where I live. My passion lies in community engagement, trying to build bridges across socioeconomic boundaries that were the source of the Cincinnati riots. I also teach step aerobics, play in a golf and broomball sports league, and am happy to share my personal experiences with you if you contact me at sstjohn@cinci.rr.com.

I had the fortune of living with a math major, a mechanical engineer, a computer scientist, and an electrical engineer for two years. This sort of cross discipline exposure helped me appreciate the similarity of the solutions. That may seem like an odd statement, but so many of the problems that my contemporaries and I faced could be modeled using similar fundamental equations that success was driven by those who had the greatest familiarity with the tools to process the data. Mathematics is the underlying tool used to understand relationships spanning topics from Nicomachean Ethics by Aristotle to multivariate data analysis with multiple bases; computers are the workhorses that allow us to analyze the data. Understanding how to build the computational tools separates a good engineer from a great engineer.

Figure 7 Sam on the Great Wall

Let's frame some of my contemporary challenges in the trials of an engineering education. My background is chemical engineering; I currently use my education to develop upstream whitening technologies for the Crest brand. One of the benefits of working at P&G is that I am afforded the opportunity to return to Clarkson to recruit fellow alumni to work at P&G. On nearly every return visit I am asked by students or professors what classes have served me best, what was lacking, or what would have been useful? I have one consistent message that I truly believe – modern engineers must have three definite strengths related to computational mathematics: 1) engineers must understand underlying equations governing the first principles of mechanical and chemical systems; 2) engineers must understand how to interpret experimental data and relate them to first principles; and 3) engineers must be able to craft the tools using mathematical modeling/computational software to design data collection system, interpret experimental data, or make mathematical extrapolations to first principles. There can no longer be two separate schools of thought keeping algebraic analysis and numerical analysis separate; they must be taught conjoined. Now, the above discourse may seem generic enough to encompass the entirety of an undergraduate engineering education. That conclusion is simply not true – too few students take the key courses to be able to function across the three criteria I have listed. These courses include (in addition to the standard engineering curricula): applied linear algebra, applied statistics and linear regression analysis, MATLAB/C++ introduction/intermediate computer programming, boundary value problems, and computational logic. The value of a rigorous mathematics and numerical computation minor cannot be understated. Let me use two examples requiring knowledge spanning the three categories that I encountered within 18 months of working at P&G.

The most recent example is one where I had to compare data collected using the same experimental design but collected at two different times. I was collecting information on the absolute rate of surface roughness change of an enamel substrate. The rate data are non-linear in time and naturally contain some experimental variation. The rates of the control groups between the two runs were different enough to prevent direct comparison of the experimental treatments without any data manipulation. Admittedly, this is a relatively simple problem to solve, but difficult to recognize without the proper tools. First, because the rates are non-linear, the data had to be transformed – in this case, a power transformation on time allowed the linearization of the data. Using the rate of roughness change of the two control groups from the two different experimental runs, the information existed to transform the basis of the first experiment into that of the second (or vice versa) while maintaining the scale of the experimental groups. I was able to recognize the solution to the problem by blending my linear algebra experience with engineering statistics. Linear algebra is not a class typically taken by engineering students, but is absolutely vital when transforming data gathered using different experimental methods or comparing data from different experimental runs. This simple problem can become significantly more complicated if a high order design of experiments is used to investigate multiple phenomena. In these cases, it is vital to understand data transformation to make clear comparisons across the entire DOX. Alternatively, if the data set were larger than the 4 samples I had, it simply becomes more efficient for the engineer to develop an algorithm to handle the data processing.

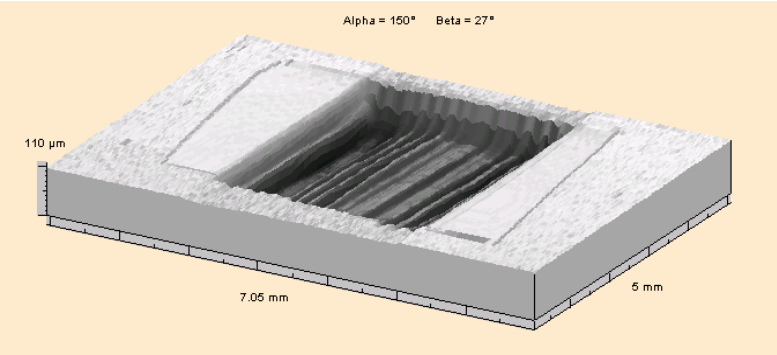

Figure 8 Abraded substrate with two control portions to the right and left of the abraded channel

In a second example, I had to design a system to compare the amount of surface wear generated on a surface by an abrasive in terms of material lost by mechanical removal. The output data from the measurement tool was simply a three dimensional map of a surface in an array of x, y, and z, coordinates. I eventually solved the problem by maintaining two control surfaces alongside an abraded region. From these control surfaces, I calculated the volume of the material removed and the average material removed is reported as a step change down from the control surface. Over the course of 4 months, I had

to process nearly 500 samples, each sample taking approximately 10 minutes to process after I developed a computer program to handle the data analysis. Using MATLAB, I wrote my own program to map the bounds of the region of the abraded portion, determine the volume of the material removed in the abraded region, and report the numerical error associated with the numerical integration technique used. This programming knowledge alone was not sufficient; the problem also required a rigorous understanding of the experimental setup, the detection resolution of the measuring system, and the ability to program, setup, and write an effective and efficient computer algorithm, as each sample contained nearly 100,000 sets of surface coordinates.

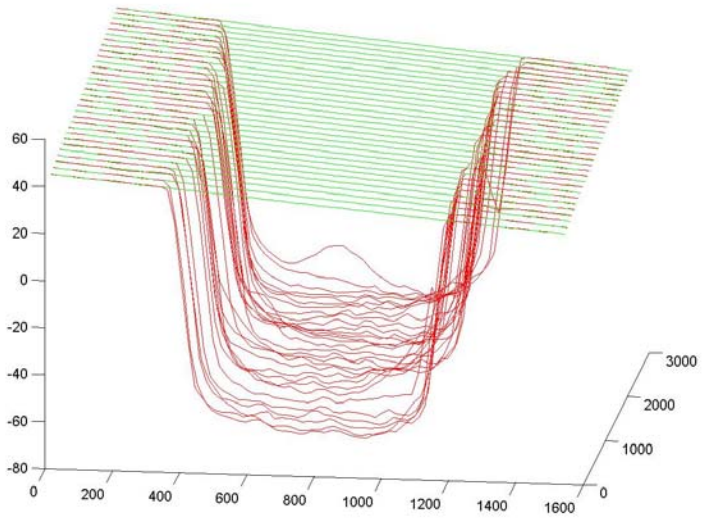

Figure 9 Matlab surface wire-frame. The region of integration was determined automatically by the computer algorithm, and the volume was determined by numerical integration between the green surface and the red substrate surface map.

The brilliant thing about computing today is that the solutions to complex technical problems are in reach of the undergraduate engineer if that engineer has the proper education to frame the experiment and the understanding of the tools to analyze the data effectively. It is simply inexcusable for an engineer today not to be able to program in MATLAB, Maple, Fortran, or similar languages or tools to solve real problems. The power of computers today is that they afford us the ability to examine complex multivariate experimental designs. These problems often do not have "black box" or "off the shelf" software solutions. The available mathematical software packages must be manipulated. Many companies turn to new engineers who have a fresh working knowledge of the latest computational packages to solve these problems. In a world of shrinking timelines where speed to market is a driving force behind innovation,

the engineer must be able to quickly manipulate the tools necessary to solve the problems of the moment while staying fluent in the language of the problems of the future.

6 Conclusion and Recommendations

6.1 Broadening the CSE Student Experience

It is *absolutely essential* that interdisciplinary collaboration be an integral part of the curriculum and the thesis research. There are a number of ways in which this can be achieved:

- Courses with multidisciplinary projects and presentations whenever possible.
  - Include different computing environments
- Participation in a multidisciplinary research team.
- Internship at a National Laboratory, a Research Experience for Undergraduates (REU) program, or in industry.
- Conference sessions – discuss growth of undergraduate paper sessions and computational REUs

The NSF National Partnership for Advanced Computational Infrastructure (NPACIs) in the U.S. have programs in place to assist in CSE education. Two excellent online news magazines provide up-to-date summaries of the activities of the NPACIs: http://www.npaci.edu/npaci/online/ and the National Computational Science Alliance (NCSA) http://www.ncsa.uiuc.edu/access/. The focused education programs are summarized through the cooperative EOT-PACI (Education, Outreach and Training) Web Site for both PACIs http://www.eot.org/. There are REU opportunities coordinated through the EOT-PACI and the details for applying are provided.

6.2 Role of SIAM

We close with some recommendations regarding the potential (and continuing) role and contributions of SIAM in CSE education and research.

1. Define the core areas and scope of this field
2. Help educate potential employers and managers on the nature and benefits of CSE
3. Outline ideas for curriculum
   - Essential courses

- Desirables
- "External fields" dependent on local conditions

4. Hold conferences on CSE including
   - CSE Undergraduate Education issues
   - Undergrad papers
5. Expand the Activity Group on CSE
   - Encourage the participation of computational researchers from science and engineering, and collaboration with their professional societies
   - Add resources page for student projects
6. Examine SIAM's existing journals to determine whether there is a place for CSE research
   - Is there a place for an undergrad papers/projects (electronic) journal?
7. Create an electronic CSE Bulletin Board
   - Discussion forum
   - Pointers to graduate degree programs
   - Infrastructure for universities, government and industry to post internship opportunities and for students to post resumes
   - Infrastructure for universities, government and industry to post job opportunities and for job seekers to post resumes
8. Publish information of use for CSE education, for example articles specifically oriented to teaching in CSE programs or courses
9. Publish CSE textbooks
10. Publish CSE research books

References

[1] SIAM Working Group on CSE Education (Linda Petzold, Chair) *Graduate Education in CSE*, SIAM Review 43 (2001) 163-177

[2] G.O.Fowler & P.R.Turner, *A special course in analysis of numerical and statistical data for advanced undergraduates*, Frontiers in Education, Boston, November 2002, p.F4H-15

[3] Geoffrey C. Fox *Remarks on Academic Programs in Computational Science*, presentation at University of Illinois, May 1995 `http://www.npac.syr.edu/users/gcf/compsci95/`

[4] W. J. Kaufmann III and L. L. Smarr, Supercomputing and the Transformation of Science, Scientific American Library, 1993.

[5] Krell Institute *Computational Science Learning Center,* `http://www.krellinst.org/learningcenter/`

[6] Rubin H. Landau, *A First Course in Scientific Computing: Symbolic, Graphic, and Numerical Modeling*, Princeton University Press, Princeton, 2005

[7] Rubin H. Landau, *Computational Physics: a Better Model for Physics Education?* Computation in Science & Engineering 8 (2006) 22-30

[8] James S. Langer, Chair, National Workshop on Advanced Scientific Computing, National Academy of Sciences, 1998.

[9] Chronicle of Higher Education, *Presidential Panel Recommends Steps to Promote Computational Science,* Chronicle of Higher Education, April 15, 2005

[10] W. Schiehlen, editor, Multibody Systems Handbook, Springer-Verlag 1990.

[11] Angela B. Shiflet and George W. Shiflet, *Introduction to Computational Science: Modeling and Simulation for the Sciences*, Princeton University Press, Princeton, 2006

[12] Lynn Arthur Steen (Ed) *Math & Bio 2010: Linking Undergraduate* D*isciplines,* Mathematical Association of America, 2005

[13] Kris Stewart, Roscoe Giles & Ilya Zaslavsky *Super-Partnerships: Computational Science Curricula, High Performance Computing and the Professional Organizations*, presented at EDUCAUSE '99, `http://www.educause.edu/`

[14] C.D. Swanson, *Computational Science Education*, Krell Institute 2003, `http://www.krellinst.org/learningcenter/CSE_survey_Nov_2003.pdf`

[15] Peter R. Turner, *Teaching scientific computing through projects*, J of Eng. Ed. 90 (2001), pp.79-83.

[16] O. Yasar & R.H. Landau, *Elements of Computational Science and Engineering Education*, SIAM Review 45 (2003) 787-805

General Publications

[17] *Chemical & Engineering News*, May 1997.

[18] *Computing in Science & Engineering*, IEEE Computer Society and American Institute of Physics, http://www.computer.org/cise/

[19] *Frontiers in Education* Internet Clearing House for FIE Conferences, http://fie.engrng.pitt.edu/

[20] *Information Week* InformationWeek.com

[21] *Scientific Computing World*, Europa Science Ltd, http://www.scientific-computing.com/

[22] *SIAM News*, Society for Industrial and Applied Mathematics, http://www.siam.org/siamnews/

[23] *SIGCSE* ACM Special Interest Group on Computer Science Education http://www.sigcse.org/

Some Undergraduate CSE Programs and their URLs

[24] Colorado School of Mines, Robert J. Kee website, http://www.mines.edu/Academic/eng/faculty/rkee/.

[25] University of Colorado, Dept. of Chemical Engineering, Victor H. Barocas website, http://spot.colorado.edu/ chemeng/faculty/barocas.html.

[26] Computational Science and Engineering, University of Illinois at Urbana-Champaign, MS/Ph.D Option handbook, M. T. Heath, Director, 1998-99. University of Urbana-Champaign CSE, website http://www.cse.uiuc.edu/programs/index.html

[27] Accelerated Strategic Computing Initiative website, http://www.lanl.gov/projects/asci/.

[28] Purdue University, CSE website http://www.cse.purdue.edu/.

[29] Queen Mary Collgege, University of London, http://www.chem.qmw.ac.uk/iupac/

[30] Rensselaer Polytechnic Institute, Dept. of Biomedical Engineering, Robert L. Spilker website,
http://www.eng.rpi.edu/ spilker/research.html.

[31] Computational Science and Engineering, KTH Stockholm, website http://www.nada.kth.se/kurser/master/index-eng.html.

[32] Computational Science and Engineering, ETH Zurich, website http://www.cse.ethz.ch.

Foundations and Organizations

[33] National Computational Science Alliance http://www.ncsa.uiuc.edu/access/

[34] National Partnership for Advanced Computational Infrastructure: Online news magazines http://www.npaci.edu/npaci/online/

[35] Shodor Education Foundation http://www.shodor.org/