



Creating Symphonic-Thinking Computer Science Graduates for an Increasingly Competitive Global Environment

“Symphonic thinking is the signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers producing a unified and pleasing sound. Entrepreneurs and inventors have long relied on this ability. But today, Symphony is becoming an essential aptitude for a much wider swath of the population.”

Daniel H. Pink

A Whole New Mind: Moving from the Information Age to the Conceptual Age

Merrick Furst

Distinguished Professor and Associate Dean
The College of Computing at Georgia Tech

Richard A. DeMillo

John P. Imlay, Jr. Dean of Computing
The College of Computing at Georgia Tech

Outline

Creating Symphonic-Thinking Computer Science Graduates for an Increasingly Competitive Global Environment

I. Impetus for innovation

1. Introduction
 - 1.1 Report organization
 - 1.2. The central role of computing in the future economy
 - 1.3. What are symphonic-thinking computer science students?
2. The future of computing in a *Flat World*, with a *Whole New Mind*
 - 2.1. Global forces at play
 - 2.1.1. Outsourcing and offshoring
 - 2.1.2. Declining undergraduate computer science enrollments
 - 2.2. What it takes to compete and succeed

II. Threads™, a new paradigm in undergraduate computing education

1. Introduction
 - 1.1 Report organization
 - 1.2. The central role of computing in the future economy
 - 1.3. What are symphonic-thinking computer science students?
2. Georgia Tech's solution: Threads™
 - 2.1. Computing identities: Threads
 - 2.2. Computing trajectories: Roles
 - 2.3. International and diversity experiences
 - 2.4. Case study: Developing computer games for handheld devices
 - 2.5. Threads™ background

Appendix A. Descriptions of Individual Threads

Appendix B. Descriptions of Individual Roles

Creating Symphonic-Thinking Computer Science Graduates for an Increasingly Competitive Global Environment

I. Impetus for Innovation

“As more nations develop technologically educated work forces, innovation is no longer the exclusive purview of the United States but emerges from numerous places around the globe. America’s long-standing leadership at the high end of the economy is no longer assured as ambitious nations with larger populations enter the innovation space with their sights focused on competition.”

“How can we prepare our graduates not only to thrive in the innovation economy, but also to be sensitive to the human and environmental impacts of innovation and to manage the disruption it causes? How can we prepare them to be lifelong learners, comfortable with the notion that their careers will likely change course several times along the way? How can we prepare them to be citizens of the world who understand the global dynamics of our economy?”

—G. Wayne Clough, President, Georgia Institute of Technology, State of the Institute 2005

“There is only one message: You have to constantly upgrade your skills. There will be plenty of good jobs out there in the flat world for people with the knowledge and ideas to seize them.”

Thomas L. Friedman, *The World is Flat: A Brief History of the Twenty-first Century*

“Symphonic thinking is the signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers producing a unified and pleasing sound. Entrepreneurs and inventors have long relied on this ability. But today, Symphony is becoming an essential aptitude for a much wider swath of the population.”

Daniel H. Pink, *A Whole New Mind: Moving from the Information Age to the Conceptual Age*

1. Introduction

The College of Computing at Georgia Tech has released a two-part report exploring the need for a new approach to undergraduate computing education in the face of an increasingly competitive global environment and describing the College’s new undergraduate computing program: Threads™. The report, “Creating symphonic-thinking computer science graduates for an increasingly competitive global environment,” presents: (1) the *problem* faced by computer science undergraduates, departments and colleges of computing, and relevant industry, and (2) the innovative *solution* developed at Georgia Tech.

The report argues that tomorrow’s undergraduate computer science students will need to develop new skills to compete successfully in the future environment, and the idea of *symphonic-thinking* is of central importance to the tremendous opportunities and challenges faced by U.S. computer science graduates. Symphonic-thinking refers to the “signature ability of composers and

conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers and producing a unified and pleasing sound ... recognizing patterns, crossing boundaries to uncover hidden connections, and making bold leaps of imagination.”¹ Symphonic-thinking students of computing will develop expertise in multiple, high-value areas of computing and act as innovative boundary crossers.

1.1. Report organization

Part I – *Impetus for innovation* – provides an introduction to the increasingly competitive global environment. The dot-com bust and resultant loss of jobs in information technology and other high-tech industries, trends in outsourcing and offshoring, and decreased enrollments in some of the nation’s top computer science undergraduate programs have forced universities, industry, and government policymakers to think seriously about whether there is a disconnect between the old way of preparing students and the new, flat world described by journalist Thomas Friedman.²

Part II – *Threads™, a new paradigm in undergraduate computing education* – describes the new approach to computer science education developed by The College of Computing at Georgia Tech – an exciting, flexible approach that is uniquely tuned to lifelong value creation and to students’ career goals and needs.

1.2. The central role of computing in the future economy

With widespread media coverage of outsourcing and offshoring to India, China, and other countries with large, growing populations of well-educated computer scientists and engineers, one might be forgiven for dismissing the central role of the U.S. computing workforce on the future. However, U.S. computing will undoubtedly drive continued technological development, scientific discovery, medical improvements, media and entertainment innovations, and economic growth across diverse business areas. Students trained in computer science – and, more broadly, in computing – will be among the change agents responsible for forging new computing breakthroughs and new interactions with other disciplines.

At the same time, it has become increasingly clear that the old way of doing things is not going to work. Old-school software engineers and computer programmers trained using traditional undergraduate computer science curricula will find it difficult to differentiate themselves and remain competitive in an increasingly global, interconnected economy. The new breed of computing graduates must adapt their views of, and integration into, the world to be successful. In a pioneering effort, The College of Computing at Georgia Tech, one of the few college-level computing programs in the nation, has been working over the past several years to develop a paradigm-changing undergraduate computing program: *Threads™*, described in detail in Part II of this report.

Threads™ promises to empower students – giving them the power to define their computing

¹ Daniel Pink, *A Whole New Mind: Moving from the Information Age to the Conceptual Age*, Riverhead Books: New York, 2005, p. 126.

² In his recent book: Thomas Friedman, *The World is Flat: A Brief History of the Twenty-first Century*, Farrar, Straus and Giroux: New York, 2005.

identities, take ownership of their career trajectories, and pursue enhanced international and diversity experiences. Threads™ graduates will lead the future of computing and its interactions with ubiquitous other areas and will graduate able to continuously add value throughout their careers, be adaptive to changing information technology value propositions, avoid being outsourced, and possess a global mindset.

1.3. What are symphonic-thinking computer science students?

Symphonic-thinking has nothing to do with loading up students' iPods with Beethoven, Bach, and Mozart to listen to while they write code. Instead, as introduced above, *symphonic-thinking* refers to the “signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers and producing a unified and pleasing sound...recognizing patterns, crossing boundaries to uncover hidden connections, and making bold leaps of imagination.” (Pink, p. 126.)

Whereas being really good at a particular aspect of computing was of tremendous value in the past, tomorrow's computer science graduates must bring together many aptitudes and skills in order to drive real, sustainable value creation. Students will have to go beyond “playing a single instrument” well to pulling together multiple, diverse computing skills into a “symphony.” Symphonic-thinking students of computing must develop expertise across an array of synergistic sub-disciplines and act as innovative boundary crossers – i.e., “see relationships the rest of us never notice. Such ability is at a premium in a world where specialized knowledge work can quickly become routinized work—and therefore be automated and outsourced away.” (Pink, p. 131.)

“Symphony ... is the ability to put together the pieces. It is the capacity to synthesize rather than to analyze; to see relationships between seemingly unrelated fields; to detect broad patterns rather than to deliver specific answers; and to invent something new by combining elements nobody else thought to pair.” (Pink, p. 126.)

“Symphony is largely about relationships. People who hope to thrive in the Conceptual Age must understand the connections between diverse, and seemingly separate, disciplines.” (Pink, p. 130.)

Along these lines, “Seeing the big picture is fast becoming a killer app in business. While knowledge workers of the past typically performed piecemeal assignments and spent their days tending their own patch of a larger garden, such work is now moving overseas or being reduced to instructions in powerful software. As a result, what has become more valuable is what fast computers and low-paid overseas specialists cannot do nearly as well: integrating and imagining how the pieces fit together.” (Pink, p. 137.)

Georgia Tech's Threads™, described in Part II of this report, enables students to create the symphony of high-value computing skills and capabilities needed to compete and succeed in the world.

2. The future of computing in a *Flat World*, with a *Whole New Mind*

Two recent, popular books – Thomas Friedman’s *The World is Flat* and Daniel Pink’s *A Whole New Mind* – present a unique view of the current and emerging globally competitive landscape and offer a compelling picture of what it will take for individuals (and organizations) to succeed going forward. Friedman’s arguments revolve largely around the idea that the global competitive playing field is being leveled. Pink’s arguments are well introduced by the following excerpt:

“The last few decades have belonged to a certain kind of person with a certain kind of mind—computer programmers who could crank code, lawyers who could craft contracts, MBAs who could crunch numbers. But the keys to the kingdom are changing hands. The future belongs to a very different kind of person with a very different kind of mind—creators and empathizers, pattern recognizers, and meaning makers. These people—artists, inventors, designers, storytellers, caregivers, consolers, big picture thinkers—will now reap society’s richest rewards and share its greatest joys.” (Pink, p. 1.)

Similar economic landscapes are described by both authors – quickly bringing the reader up to speed on the urgency of focusing on what it takes to compete and succeed in Friedman’s *Globalization 3.0* and Pink’s *Conceptual Age*.

“... a new, flatter, global playing field. As this new playing field became established, both businesses and individuals began to adopt new habits, skills, and processes to get the most out of it. They moved from largely vertical means of creating value to more horizontal ones. The merger of this new playing field for doing business with the new ways of doing business was the second convergence, and it actually helped to flatten the world even further. Finally, just when all of this flattening was happening, a whole new group of people, several billion, in fact, walked out onto the playing field from China, India, and the former Soviet Empire.” (Friedman, p. 175.)

“In recent years, few issues have generated more controversy or stoked more anxiety than outsourcing. ... [P]rogrammers ... throughout India, the Philippines, and China are scaring the bejeezus out of software engineers and other left-brain professionals in North America and Europe, triggering protests, boycotts, and plenty of political posturing. The computer programming they do, while not the most sophisticated that multinational companies need, is the sort of work that until recently was done almost exclusively in the United States—and that provided comfortable white-collar salaries of upward of \$70,000 a year.” (Pink, p. 37.)

2.1. Global Forces at Play

Every two years, the National Science Foundation¹ (NSF) publishes a set of science and engineering (S&E) indicators for the U.S. The most recent edition includes the following introduction to the status of the global forces at play affecting science and engineering students and graduates:

“The international S&E labor force is growing and becoming increasingly mobile. Governments are implementing policies designed to lure more of their citizens into S&E; keep their researchers at home, or in the case of the EU, in EU countries; and attract highly trained S&E personnel from abroad. Private firms are responding to competitive pressures and market opportunities by opening high-technology operations in foreign locations, developing strategic international alliances, and consummating cross-national spinoffs and mergers. A consequence of these trends

¹ The National Science Foundation (NSF) is a federal agency that provides funding for approximately 20% of federally-supported basic research conducted by America’s colleges and universities (<http://www.nsf.gov>). In many fields such as mathematics, computer science, and the social sciences, NSF is the major source of federal backing.

is the further spread of technological know-how and the development of significant scientific and technical capacity in new locations across the globe.”¹

Two key, ongoing trends provide impetus for the innovation of Threads™: outsourcing/offshoring and computer science enrollment patterns.

2.1.1. Outsourcing and offshoring

Outsourcing and offshoring of information technology jobs are neighboring trends that have received intense media coverage. Outsourcing refers to the transfer of key activities and jobs to another organization; offshoring refers to the transfer of activities and jobs important to an organization to an overseas location. According to Pink:

“Much of the anxiety over this issue outstrips the reality. We are not all going to lose our jobs tomorrow. Outsourcing is overhyped in the short term. But it’s underhyped in the long term. As the cost of communicating with the other side of the globe falls essentially to zero, and as developing nations continue to mint millions of extremely capable knowledge workers, the working lives of North Americans, Europeans, and Japanese people will change dramatically.”
(Pink, p. 39.)

In the height of the dot-com bust, U.S. firms discarded large numbers of information technology jobs. One study in 2002, from the Information Technology Association of America, reported that over 500,000 U.S. information technology workers lost their jobs in 2001.² Another study by the same organization found that from 2000 to early 2004, “approximately 100,000 computer software and services jobs have moved offshore, including both domestic jobs eliminated and new jobs created overseas.”³ According to a study by Forrester Research, a predicted 3.3 million white-collar jobs will be offshored by 2015, representing \$136 billion in lost U.S. wages. (AAAS, p. 16.) Furthermore, “this forecast included 473,000 IT positions over and above those already lost to earlier movements of this kind.” (Ibid.) Finding data to support his claim that outsourcing and offshoring are overhyped in the short term but underhyped in the long term, Pink reports that, “one out of ten jobs in the U.S. computer, software, and information technology industry will move overseas in the next two years. One in four IT jobs will be offshored by 2010.” (Pink, p. 39.)

It is difficult to overstate the rapid and continued growth of India as a computing services powerhouse. As of March 2004, 60% of the information technology jobs offshored by the U.S. went to India; Ireland, China, and Canada each roughly received 10% of U.S. offshored information technology jobs.⁴ The following data provide a picture of India’s role in the global information technology industry:

-
- 1 National Science Board (NSB), *Science and Engineering Indicators 2004*, National Science Foundation, p. 0-3.
 - 2 American Association for the Advancement of Science (AAAS), *Bringing Women and Minorities into the IT Workforce: The Role of Nontraditional Educational Pathways*, 2005, p. 10.
 - 3 Information Technology Association of America (ITAA), “Adding Value...Growing Careers: The Employment Outlook in Today’s Increasingly Competitive IT Job Market,” September 2004, p. 5.
 - 4 Alok Aggarwal, “Moving Up the Value Chain,” presentation at 2004 Computing Research Association (CRA) Conference at Snowbird, July 2004.

- Approximately 350,000 engineering graduates are produced by colleges and universities in India each year. (Pink, p. 37.)
- Of these, approximately 120,000 specialized in information technology (versus 75,000 U.S. engineering graduates that specialize in information technology disciplines). (Aggarwal.)
- A reported 1.45 million information technology professionals will work in India in 2010, up from 656,000 in 2003, and the share of these individuals providing offshore services is predicted to double, yielding 860,000 Indian information technology professionals providing offshore services in 2010 with an estimated \$37.5 billion in export revenue. (Ibid.)
- India's software and services exports increased by more than 34% from 2004 to 2005.¹

The statistics for individual companies outsourcing information technology jobs to India and other countries are no less impressive: (Pink, pp. 37-38.)

- More than half of Fortune 500 companies outsource software work to India.
- Almost 50% of GE's software is developed in India, where the company employs 20,000 people.
- Hewlett-Packard employs several thousand software engineers in India.
- Siemens employs 3,000 computer programmers in India and is moving another 15,000 such jobs overseas.
- Motorola, Nortel, and Intel operate software development centers in Russia.
- Electronic Data Systems has software developers in Egypt, Brazil, and Poland.

Additionally, in December 2005, Microsoft announced that, over the next four years, it will double its workforce in India (to 7,000), grow its R&D investment in the country by \$1.7 billion, and add a second R&D center in Bangalore, to complement its existing one in Hyderabad. (Bagla, p. 1754.)

Accompanying the outsourcing/offshoring of information technology activities and jobs to India and other countries with large, growing populations of well-educated computer scientists and engineers is a movement up the value chain. As Friedman reports, "the globalization of innovation' [represents] an end to the old model of a single American or European multinational handling all the elements of the development product cycle from its own resources. More and more American and European companies are outsourcing significant research and development tasks to India, Russia, and China." (Friedman, p. 30.) While past outsourced information technology services included application development and maintenance, simple coding and porting from one language to another, and fixing the Y2K problem, current and near future outsourced services include information technology consulting, product development, R&D, and engineering design services. (Aggarwal.)

1 Pallava Bagla, "Booming Computer Science Sector Seen as a Mixed Blessing," *Science*, v. 310, 16 December 2005, p. 1754.

2.1.2. Declining undergraduate computer science enrollments

Undergraduate enrollments and degree production in computer science have shown a dramatic up-and-down pattern over the past 25 years. The number of undergraduate computer science degrees increased significantly from 1981 to 1986 and decreased significantly from 1987 to 1997. (AAAS, pp. 3-4.) Enrollments peaked during the dot-com boom of the late 1990s and have fallen since the bust in 2001. An August 2005 *New York Times* article reported that “the number of students choosing computer science as a major is 39 percent lower than in the fall of 2000, the last of the dot-com bubble years.”¹

In the summer of 2004, an article on *CNET News* provided the following snapshot of undergraduate computer science programs: MIT’s electrical engineering and computer science department – new undergraduate majors down to under 200, from 240 in the previous year; Carnegie Mellon’s school of computer science – applications down to 2,000, from 3,200 in 2001; University of California at Berkeley – undergraduate computer science majors down to 226, from 260 in the spring of 2003; and Stanford University – undergraduate computer science majors down to 118 in the past year, from 171 in 2000-2001.²

Some have linked the decline in undergraduate computing program enrollments to negative student and parent perceptions, especially given the prominent media reports of outsourcing, offshoring, and the resulting current and future job losses.³ Along these lines is the realization that a degree in computer science no longer offers the lure of guaranteed employment that it once seemed to offer. Finally, computing has become so interdisciplinary and so linked with other disciplines – within engineering and science and beyond – that students interested in computing no longer have to major in computer science to explore programming and modeling. Indeed, “the march of computing is rippling across all academic disciplines. Even as computer science students are being encouraged to take more courses outside their major, students in other disciplines are finding more often that they need to use, design and sometimes write computer programs.” (Lohr.)

Clearly, the global environment is rapidly changing the playing field for computing jobs in the U.S. Nevertheless, the predicted demand for computing jobs in this country is robust. The latest predictions from the Bureau of Labor Statistics (from 2001) show an approximate 15% increase in U.S. jobs from 2000-2010, with S&E jobs predicted to grow 47% (representing 2.2 million new scientists and engineers). (NSB, p. 3-7.) Growth in computer-related S&E jobs is predicted to represent approximately 86% of the growth across all S&E occupations. (Ibid.) Drilling deeper, “the number of jobs for computer software engineers is expected increase from 697,000 to 1.4 million and employment for computer systems analysts is expected to grow from 431,000 to 689,000 jobs.” (Ibid.)

As reported in the *New York Times*, “for people who stay in computing, the job outlook is

¹ Steve Lohr, “A Technie, Absolutely, and More,” *The New York Times*, 23 August 2005.

² Ed Frauenheim, “Students saying no to computer science,” *CNET News*, 11 August 2004.

³ Stephen Seidman, “Software Offshoring – Risks and Opportunities for Computing Programs,” presentation at 2004 CRA Conference at Snowbird, July 2004.

brightest for those skilled in the application of technology. While jobs in categories like programming have declined since 2000, according to the Labor Department, the need for information technology experts has not.” (Lohr.)

2.2. What it takes to compete and succeed

An underlying goal for undergraduate computer science students and graduates entering the global economy is to become, as Friedman puts it, untouchable – i.e., “people whose jobs cannot be outsourced.” (Friedman, p. 238.) As you get closer to the user, application, and point of innovation, it becomes harder to be outsourced. Undergraduate computer science graduates must become: (1) extremely adaptable; (2) apt at forging new and dynamic relationships, tackling novel challenges, and synthesizing the “big picture;” and (3) more competent at utilizing creativity and tacit knowledge. Unfortunately, the current way of teaching undergraduates in computer science is not well-tuned to this goal or to the increasingly interconnected, global economy generally.

Pink encourages “forcing knowledge workers in the advanced world to master abilities that can’t be shipped overseas” (Pink, p. 46.) and offers the following checklist:

“To survive in this age, individuals and organizations must examine what they’re doing to earn a living and ask themselves three questions:

1. Can someone overseas do it cheaper?
2. Can a computer do it faster?
3. Is what I’m offering in demand in the age of abundance?” (Pink, p. 51.)

Expertise in computer programming is clearly no longer enough to be competitive. As reported recently in the *New York Times*, “expanding [students’] expertise beyond computer programming is crucial to future job security as advances in the Internet and low-cost computers make it easier to shift some technology jobs to nations with well-educated engineers and lower wages, like India and China.” (Lohr.)

According to Friedman, individuals must “want constantly to acquire new skills, knowledge, and expertise that enable you constantly to be able to create value.” (Friedman, p. 239.) Indeed, “being adaptable in a flat world, knowing how to ‘learn how to learn,’ will be one of the most important assets any worker can have, because job churn will come faster, because innovation will happen faster.” (Ibid.) According to the Information Technology Association of America, “for the IT worker interested in moving a career forward, problem-solving (and value creation) must be considered both a matter of having up-to-date technical skills, but also being able to step back and see the organization’s bigger business picture.” (ITAA, p. 16.) As Pink describes:

“[T]oday’s knowledge workers will likewise have to command a new set of aptitudes. They’ll need to do what workers abroad cannot do equally well for much less money ... forging relationships rather than executing transactions, tackling novel challenges instead of solving routine problems, and synthesizing the big picture rather than analyzing a single component.” (Pink, pp. 39-40.)

“[A]s the scut work gets off-loaded, engineers and programmers will have to master different aptitudes, relying more on creativity than competence, more on tacit knowledge than technical manuals, and more on fashioning the big picture than sweating the details.” (Pink, pp. 44-45.)

In short, to compete and succeed in the increasingly competitive global economy, students of computing must become symphonic-thinking. They must develop and wrangle expertise in multiple spheres; recognize and synthesize patterns; cross boundaries; be able to corral diverse technologies, disciplines, individuals, and organizations and produce a unified, value-added solution; and make bold leaps toward innovating and inventing something new from previously uncombined elements.

The need for symphonic-thinking computing graduates—and requisite changes to undergraduate computing curricula and programs – has reached a tipping point. Georgia Tech’s Threads™, described in detail in Part II of this report, is a new approach to computer science education – an approach that is more flexible, exciting, and tuned to career goals and needs.

Creating Symphonic-Thinking Computer Science Graduates for an Increasingly Competitive Global Environment

II. Threads™, a New Paradigm in Undergraduate Computing Education

“As more nations develop technologically educated work forces, innovation is no longer the exclusive purview of the United States but emerges from numerous places around the globe. America’s long-standing leadership at the high end of the economy is no longer assured as ambitious nations with larger populations enter the innovation space with their sights focused on competition.”

“How can we prepare our graduates not only to thrive in the innovation economy, but also to be sensitive to the human and environmental impacts of innovation and to manage the disruption it causes? How can we prepare them to be lifelong learners, comfortable with the notion that their careers will likely change course several times along the way? How can we prepare them to be citizens of the world who understand the global dynamics of our economy?”

G. Wayne Clough, President, Georgia Institute of Technology, *State of the Institute 2005*

“There is only one message: You have to constantly upgrade your skills. There will be plenty of good jobs out there in the flat world for people with the knowledge and ideas to seize them.”

Thomas L. Friedman, *The World is Flat: A Brief History of the Twenty-first Century*

“Symphonic thinking is the signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers producing a unified and pleasing sound. Entrepreneurs and inventors have long relied on this ability. But today, Symphony is becoming an essential aptitude for a much wider swath of the population.”

Daniel H. Pink, *A Whole New Mind: Moving from the Information Age to the Conceptual Age*

1. Introduction

The College of Computing at Georgia Tech has released a two-part report exploring the need for a new approach to undergraduate computing education in the face of an increasingly competitive global environment and describing the College’s new undergraduate computing program: Threads™. The report, “Creating symphonic-thinking computer science graduates for an increasingly competitive global environment,” presents: (1) the *problem* faced by computer science undergraduates, departments and colleges of computing, and relevant industry, and (2) the innovative *solution* developed at Georgia Tech.

The report argues that tomorrow’s undergraduate computer science students will need to develop new skills to compete successfully in the future environment, and the idea of *symphonic-thinking*

is of central importance to the tremendous opportunities and challenges faced by U.S. computer science graduates. Symphonic-thinking refers to the “signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers and producing a unified and pleasing sound ... recognizing patterns, crossing boundaries to uncover hidden connections, and making bold leaps of imagination.”¹ Symphonic-thinking students of computing will develop expertise in multiple, high-value areas of computing and act as innovative boundary crossers.

1.1. Report organization

Part I – *Impetus for innovation* – provides an introduction to the increasingly competitive global environment. The dot-com bust and resultant loss of jobs in information technology and other high-tech industries, trends in outsourcing and offshoring, and decreased enrollments in some of the nation’s top computer science undergraduate programs have forced universities, industry, and government policymakers to think seriously about whether there is a disconnect between the old way of preparing students and the new, flat world described by journalist Thomas Friedman.²

Part II – *Threads™, a new paradigm in undergraduate computing education* – describes the new approach to computer science education developed by The College of Computing at Georgia Tech – an exciting, flexible approach that is uniquely tuned to lifelong value creation and to students’ career goals and needs.

1.2. The central role of computing in the future economy

With widespread media coverage of outsourcing and offshoring to India, China, and other countries with large, growing populations of well-educated computer scientists and engineers, one might be forgiven for dismissing the central role of the U.S. computing workforce on the future. However, U.S. computing will undoubtedly drive continued technological development, scientific discovery, medical improvements, media and entertainment innovations, and economic growth across diverse business areas. Students trained in computer science – and, more broadly, in computing – will be among the change agents responsible for forging new computing breakthroughs and new interactions with other disciplines.

At the same time, it has become increasingly clear that the old way of doing things is not going to work. Old-school software engineers and computer programmers trained using traditional undergraduate computer science curricula will find it difficult to differentiate themselves and remain competitive in an increasingly global, interconnected economy. The new breed of computing graduates must adapt their views of, and integration into, the world to be successful. In a pioneering effort, The College of Computing at Georgia Tech, one of the few college-level computing programs in the nation, has been working over the past several years to develop a paradigm-changing undergraduate computing program: *Threads™*, described in detail in Part II of this report.

1 Daniel Pink, *A Whole New Mind: Moving from the Information Age to the Conceptual Age*, Riverhead Books: New York, 2005, p. 126.

2 In his recent book: Thomas Friedman, *The World is Flat: A Brief History of the Twenty-first Century*, Farrar, Straus and Giroux: New York, 2005.

Threads™ promises to empower students – giving them the power to define their computing identities, take ownership of their career trajectories, and pursue enhanced international and diversity experiences. Threads™ graduates will lead the future of computing and its interactions with ubiquitous other areas and will graduate able to continuously add value throughout their careers, be adaptive to changing information technology value propositions, avoid being outsourced, and possess a global mindset.

1.3. What are symphonic-thinking computer science students?

Symphonic-thinking has nothing to do with loading up students’ iPods with Beethoven, Bach, and Mozart to listen to while they write code. Instead, as introduced above, *symphonic-thinking* refers to the “signature ability of composers and conductors, whose jobs involve corralling a diverse group of notes, instruments, and performers and producing a unified and pleasing sound...recognizing patterns, crossing boundaries to uncover hidden connections, and making bold leaps of imagination.” (Pink, p. 126.)

Whereas being really good at a particular aspect of computing was of tremendous value in the past, tomorrow’s computer science graduates must bring together many aptitudes and skills in order to drive real, sustainable value creation. Students will have to go beyond “playing a single instrument” well to pulling together multiple, diverse computing skills into a “symphony.” Symphonic-thinking students of computing must develop expertise across an array of synergistic sub-disciplines and act as innovative boundary crossers – i.e., “see relationships the rest of us never notice. Such ability is at a premium in a world where specialized knowledge work can quickly become routinized work—and therefore be automated and outsourced away.” (Pink, p. 131.)

“Symphony ... is the ability to put together the pieces. It is the capacity to synthesize rather than to analyze; to see relationships between seemingly unrelated fields; to detect broad patterns rather than to deliver specific answers; and to invent something new by combining elements nobody else thought to pair.” (Pink, p. 126.)

“Symphony is largely about relationships. People who hope to thrive in the Conceptual Age must understand the connections between diverse, and seemingly separate, disciplines.” (Pink, p. 130.)

Along these lines, “Seeing the big picture is fast becoming a killer app in business. While knowledge workers of the past typically performed piecemeal assignments and spent their days tending their own patch of a larger garden, such work is now moving overseas or being reduced to instructions in powerful software. As a result, what has become more valuable is what fast computers and low-paid overseas specialists cannot do nearly as well: integrating and imagining how the pieces fit together.” (Pink, p. 137.)

Georgia Tech’s Threads™ enables students to create the symphony of high-value computing skills and capabilities needed to compete and succeed in the world.

2. Georgia Tech’s solution: Threads™

The underlying goal of Threads™ is to increase the value of an undergraduate computer science degree from The College of Computing at Georgia Tech – to produce graduates who will be in high demand and who will continuously contribute value throughout successful careers. An aim is to produce graduates who have a broad set of skills and are not easily outsourcable. Facing the challenges of an increasingly global economy and competitive information technology workforce, the rapid convergence of technologies, and the changing “value stack,” the College of Computing undertook a multi-year effort to develop Threads™, including interactions with faculty across the College, current students, alumni, and industry.

Threads™, described in detail below, empowers students with the directions, tools, and opportunities needed to figure out what kind of computationalists they want to become. Threads™ aims to attract a diverse undergraduate population and produce lifetime-learning graduates tuned to the future, globally competitive economy.

Threads™ consists of two macro components: (1) students’ *computing identities* – defined by two intertwined pathways (or *Threads*) through the program, and (2) students’ *computing trajectories* – by which students’ computing identities are translated into what they want to become in a functional sense (via the exploration of *Roles*).

Threads are eight sets of broad, horizontal skills that live within and outside of computing. Any two Threads can be intertwined, leading to a degree and ultimately to a student’s computing identity. In total, there are 28 possible combinations of two Threads. Students learn a robust set of technical skills and computer programming languages in all Threads, and much of the core content in each Thread overlaps. Regardless of which two Threads are combined, the synergy yields an accredited Bachelor of Science degree in computer science.

$$\text{THREAD}_1 \times \text{THREAD}_2 \rightarrow \text{B.S. degree from Georgia Tech}$$

An Example of Threads™: Computer Security Expert

Imagine a Georgia Tech undergraduate computer science student in her sophomore year interested in computer security. She might combine the INFORMATION INTERNETWORKS thread – to learn how data is stored, retrieved, encoded, and transmitted – with the PEOPLE thread – to learn how people use technology and how to run experiments with human subjects. Her trajectory through the program will include opportunities to explore future career Roles – e.g., as a master practitioner or innovator. She will craft a valuable computing identity and trajectory to become someone able to design, invent, and build secure computing systems enabling people to securely manage their information.

Information Internetworks *where computing meets data*

Prepares students for information management by helping them to capture, represent, organize, transform, manage, and present information securely and efficiently.

People *where computing meets users*

Prepares students by helping them to understand the theoretical and computational foundations for designing, building, and evaluating systems that treat the human as a central component.

2.1. Computing identities: Threads

Threads™ creates a flexible, exciting curriculum – responding to a nationwide challenge by creating much more than an altered core curriculum or CS + x program. Threads are each partial paths through the course offerings of the university, and each student constructs his or her own personalized computer science degree by weaving together two Threads. Innovative, new courses are being developed, and many existing courses will be updated in order to respond to the Threads of enrolled students. But more than innovations in courses, Threads™ establishes a new mindset for students to see the synergistic big picture behind the courses that they take.

Each Thread provides a skill set and credential basis that allows graduates to create value in ways beyond what would be possible with only a narrowly-focused tool set. A Thread provides an intuitive, flexible, and mutually-strengthening set of courses that allows a student to craft his or her own distinctive future in an area that is certain to have societal value in the emerging world.

The eight Threads currently defined by the College of Computing are:

- Computational Modeling
- Embodiment
- Foundations
- Information Internetworks
- Intelligence
- Media
- People
- Platforms

These eight Threads are briefly described below. Appendix A contains a more detailed description of each Thread.

- **Computational Modeling.** The COMPUTATIONAL MODELING thread is where computing meets and describes the world. COMPUTATIONAL MODELING prepares students with the technical knowledge and skills necessary for expressing, specifying, understanding, creating, and exploiting computational models that represent cognitive and physical processes. It prepares students for fields as diverse as artificial intelligence, machine learning, perception, cognitive science, and graphics. The student who pursues COMPUTATIONAL MODELING can combine it with EMBODIMENT to become a roboticist, or with PEOPLE to build adaptive interfaces, or with MEDIA to prepare for a career in graphics, or with ...
- **Embodiment.** The EMBODIMENT thread is where computing meets the world. It is concerned with computation that operates under possible severe physical constraints. Via EMBODIMENT one learns how to create and evaluate computational artifacts that are embedded in physical objects and interact in the physical world, typically in real time. The

student who pursues EMBODIMENT can combine it with PLATFORMS to build devices that are small, power- and CPU-limited, or COMPUTATIONAL MODELING to build autonomous robots, or with PEOPLE to study human-robot interaction, or with ...

- **Foundations.** The FOUNDATIONS thread is where computing meets itself. FOUNDATIONS teaches students the theoretical and mathematical foundations underlying a wide range of computing disciplines. The student who pursues FOUNDATIONS can combine it with EMBODIMENT in order to provide performance bounds for robotic planning algorithms, or with PLATFORMS to become a researcher in programming languages, or with ...
- **Information Internetworks.** The INFORMATION INTERNETWORKS thread is where computing meets data. Information-centric computing prepares students for information management by helping them to capture, represent, organize, transform, manage, and present information securely and efficiently. The student who pursues INFORMATION INTERNETWORKS can combine it with COMPUTATIONAL MODELING to study text retrieval and classification, or with PEOPLE to pursue research in data visualization, or with ...
- **Intelligence.** The INTELLIGENCE thread is where computing meets and models intelligence. INTELLIGENCE is concerned with computational models of intelligence from top to bottom, with an emphasis on designing and implementing artifacts that exhibit various levels of intelligence as well as understanding and modeling natural cognitive agents such as humans, ants, or bees. Students acquire the technical knowledge and skills necessary for expressing, specifying, understanding, creating, and exploiting computational models that represent cognitive processes. It prepares students for fields as diverse as artificial intelligence, machine learning, perception, and cognitive science, as well as for fields that benefit from applications of techniques from those fields. The student who pursues INTELLIGENCE can combine it with EMBODIMENT to become a roboticist, or PEOPLE to build adaptive interfaces, or with MEDIA to build smart, adaptive entertainments, or with ...
- **Media.** The MEDIA thread is where computing meets design. MEDIA prepares students by helping them to understand the technical and computational capabilities of systems in order to exploit their abilities to provide creative outlets. The student who pursues MEDIA can combine it with COMPUTATIONAL MODELING to study animation, or with INFORMATION INTERNETWORKS to high performance database systems, or with PEOPLE to explore visualization of high-bandwidth data streaming, or with ...
- **People.** The PEOPLE thread is where computing meets users. PEOPLE prepares students by helping them to understand the theoretical and computational foundations for designing, building, and evaluating systems that treat the human as a central component. The student who pursues PEOPLE can combine it with EMBODIMENT to study human-robot interaction, or with INFORMATION INTERNETWORKS to pursue research in data visualization, or with COMPUTATIONAL MODELING for learning sciences and technology, or with PLATFORMS to explore ubiquitous computing, or with ...

- **Platforms.** The PLATFORMS thread is where many of the practical skills of computing are learned. Like the FOUNDATIONS thread, PLATFORMS lies at the center of computing. It prepares students to create and evaluate computer architectures, systems and languages across a variety of paradigms and approaches. The student who pursues SYSTEMS can combine it with FOUNDATIONS to study distributed high performance computing algorithms, or with INFORMATION INTERNETWORKS to study real-time data retrieval systems, or with PEOPLE to pursue research in developing programming environments, or with ...

While one cannot expect high school students to necessarily know what a network engineer is or whether they want to become one, they understand, for example, that they want to learn about how computers communicate over the Internet. Via Threads™, incoming students better understand how their future opportunities fit with their interests and goals and within the big picture of computer science and computing. Threads™ allows undergraduates to customize their identities as students of computing and, importantly, to finely tune their skills, experiences, and ways of thinking to the competitive global environment. Graduates of Threads™ will possess broad and dynamic skills and experiences – making them more resistant to outsourcing.

Threads are vertical concepts, not linked to particular technologies. As such, the eight Threads – and their combinations into students’ computing identities – serve to match with incoming students’ goals (e.g., an interest in computing in media). Undoubtedly, undergraduates can find making course choices intimidating, and often they just want to know from their advisors what they should take. An appeal of Threads™ to students is a dramatically increased understanding of how their courses sync together. The possibilities defined by Threads are large enough to provide diverse and dynamic opportunities for students but small enough for students to understand and get their hands around. Threads provide key selection algorithms to help students select the most relevant courses for their computing identities, instead of filing all students through a core curriculum and then providing a pool of various electives from which to choose. Students see that they are going down a path; they “see the point” and better understand how their courses fit together. Preliminary discussions with incoming freshman computer science majors about Threads have revealed that they are more or less able to understand what each Thread represents based on their titles, even though they do not yet fully grasp the finer details of the computer science and computing landscape.

Threads™ offers students the opportunity to pursue an almost limitless array of computing areas. The following list provides a sampling of these areas, followed by the two most related Threads:

- Adaptive entertainments (*INTELLIGENCE × MEDIA*)
- Adaptive interfaces (*PEOPLE × COMPUTATIONAL MODELING*)
- Animation (*MEDIA × COMPUTATIONAL MODELING*)
- Computer security (*PEOPLE × INFORMATION INTERNETWORKS*)
- Data visualization (*PEOPLE × INFORMATION INTERNETWORKS*)

- Developing computer games for handheld devices (*MEDIA × EMBODIMENT*)
- Developing programming environments (*PEOPLE × PLATFORMS*)
- Distributed high-performance computing algorithms (*PLATFORMS × FOUNDATIONS*)
- Graphics (*MEDIA × COMPUTATIONAL MODELING*)
- High-performance database systems (*MEDIA × INFORMATION INTERNETWORKS*)
- Human-robot interaction (*PEOPLE × EMBODIMENT*)
- Learning science and technology (*PEOPLE × COMPUTATIONAL MODELING*)
- Performance bounds for robotic planning algorithms (*EMBODIMENT × FOUNDATIONS*)
- Programming languages (*PLATFORMS × FOUNDATIONS*)
- Real-time data retrieval systems (*PLATFORMS × INFORMATION INTERNETWORKS*)
- Robotics (*INTELLIGENCE × EMBODIMENT*)
- Small, power- and CPU-limited devices (*EMBODIMENT × PLATFORMS*)
- Text retrieval (*COMPUTATIONAL MODELING × INFORMATION INTERNETWORKS*)
- Ubiquitous computing (*PEOPLE × PLATFORMS*)
- Visualization of high-bandwidth data streaming (*PEOPLE × MEDIA*)

The Threads™ website (<http://www.coc.gatech.edu/threads>) maintained by The College of Computing at Georgia Tech provides additional examples of computing identities that students can create by combining Threads, including bioinformatics, mobile computing, multimedia distribution, security and data extraction, and web development.

2.2. Computing trajectories: Roles

As described above, two intertwined Threads form the core of students' computing identities. The computing trajectories that students pursue with these Threads – within a rapidly changing environment in which information technology values are constantly changing – round out the Georgia Tech undergraduate computer science experience. Students' computing trajectories are defined by Roles.

Roles are integrated into all aspects of the College of Computing experience within Threads™. Roles are orthogonal to Threads; while Threads are about what students are doing (the content of their degree), Roles are about why students are doing it (how they want to apply their degree in the real world). Students pursue a rigorous, top-notch computer science undergraduate program like the one at Georgia Tech not just because they want to learn something but because they want to be something – hence the introduction of Roles.

At Georgia Tech, students all have different goals. Some want to earn a degree in mechanical engineering. Some want to become psychologists. Some want to be software engineers. Others want to become roboticists. Even the roboticists want to apply the skills they develop as

students differently. Some want to sit down on a floor and build intelligent working robots with their own hands and a screwdriver. Others want to start their own companies designing and marketing small household robots. Some may want to eventually pursue a law degree and develop guidelines for the ethical use of military robots. There are almost as many possibilities as there are students.

The four Roles currently defined by the College of Computing are MASTER PRACTITIONER, ENTREPRENEUR, INNOVATOR, and COMMUNICATOR. Similar to choosing Threads, students choose one of more Roles to explore for course credit, and these Roles help guide their course selection and direct their pursuit of the dynamic extracurricular activities available to College of Computing undergraduates. The four Roles are briefly described below. Appendix B contains a more detailed description of each Role.

- **Master Practitioner** – Expert programmer who possesses the technical skill and experiences to thoroughly design, construct, and validate computer-based systems either alone or as a part of a large team. The master practitioner is the Programmer writ large. The MASTER PRACTITIONER is not just a hacker or code monkey, but a person who can apply careful abstraction to any problem and implement a well-designed solution in one of several programming languages and styles. The MASTER PRACTITIONER is interested in the exercise and mastery of technical skill and is likely to be an employee of a corporation.
- **Entrepreneur** – Creator and leader of new enterprises that bring technology to the public at large, specifically in the form of new products and services. The model of an ENTREPRENEUR is the founder of a start-up company that provides products and services that impact or transform society.
- **Innovator** – Discoverer of new knowledge and constructor of ground-breaking solutions to problems. The model of an INNOVATOR is the academic or industrial research scientist making discoveries and inventions. These discoveries will eventually make a difference in society but result from investigations that are not guaranteed to yield practical results.
- **Communicator** – Individual capable of communicating technical information to the technologist and the layperson alike. The COMMUNICATOR is technically skilled and informed and works hard developing the communication skills necessary to effectively share their knowledge with interested but less expert audiences.

Via Roles, undergraduates get credit for exploring possible computing trajectories. A student interested in becoming a MASTER PRACTITIONER may choose to take the new Real-World Lab course (which allows students to form development teams and solve a real problem for a real customer) or the new Architecture Studio course (which provides intense involvement in practical techniques such as pair programming). A student interested in becoming an ENTREPRENEUR may take one or more courses in the School of Management and get credit for participating in the new Undergraduate Business Opportunities in Computing (UBOC) program. Via this program, interdisciplinary teams of College of Computing and College

of Management students are formed under the mentorship of local business leaders. The teams learn to develop business plans for computing enterprises and demonstrate their ideas in a competition for start-up capital. A student interested in computing research and in becoming an INNOVATOR would pursue undergraduate research opportunities with College of Computing faculty via one of several mechanisms, including an independent research project for course credit, participation in the established Undergraduate Research Opportunities in Computing (UROC) program, or by working in a professor's lab via a summer internship. A student interested in becoming a COMMUNICATOR may serve as a teaching assistant in one or more courses or carry out an internship with a campus computing group (e.g., Georgia Tech's Office of Information Technology) writing effective technical documentation or user tutorials. Of course, students may wish to define and explore more than one role – e.g., combining ENTREPRENEUR with an understanding of how to program on small devices to the same level as a MASTER PRACTITIONER.

By offering undergraduates the opportunity to explore multiple computing trajectories, Roles help to develop a culture of innovation, risk taking, and continual learning in students – attributes that increase students' value in the global economy. As a defined mechanism, Roles provide a formalized way for undergraduates to develop the essential creativity they need to successfully express their computing identities in the real world. Furthermore, via Roles, the Threads™ program hopes to increase the percentage of College of Computing undergraduates pursuing research experiences (currently 20%) and to strengthen existing linkages to other colleges at Georgia Tech (e.g., the College of Management).

2.3. International and diversity experiences

As Threads™ is implemented, it is anticipated that the innovative curriculum will more fully reward international experience and capability. The College of Computing has a dedicated goal of introducing more international experiences and a more global outlook into the program, and work is underway to accomplish this goal. An existing summer program in Spain, jointly operated with the Polytechnic Institute in Barcelona, currently enrolls over 100 students, and Georgia Tech Lorraine, the European campus of Georgia Tech in Metz, France, now offers courses from the College of Computing, bringing dual degree programs with EU universities to computing students. Furthermore, plans are underway to recognize graduates with international experience via certificates and “with international experience” designations on Georgia Tech degrees.

An additional expectation is that the implementation of Threads™ will enable the College of Computing to attract and retain a broader range of students, including larger numbers of women and underrepresented minorities, into computing and computer science. While there is some debate to the findings, research conducted in the late 1990s at Carnegie Mellon University (and published in the book, *Unlocking the Clubhouse: Women in Computing*) suggests that “Men are generally interested in computers as tools and objects of study...; women are more interested

in what computers can do for science, the arts, or society.”¹ Regardless of the ubiquity of this conclusion, by providing an entry point into the full breadth of computing and computer science and by letting students define their own trajectories through an innovative, new program, the College of Computing at Georgia Tech hopes to increase the diversity of its student body – increasing numbers of and participation from women and underrepresented minorities. Furthermore, by having students pursuing different Threads and crafting individualized computing identities taking classes together, classroom diversity and value to the students are enhanced.

Overall, Threads™ empowers the student and helps them see the big picture of their curriculum and trajectory through the undergraduate program. Students define their Threads, explore Roles, and are exposed to international and diversity experiences.

Threads™: (THREADS × ROLES) × International/diversity

2.4. Case study: Developing computer games for handheld devices

This case study provides an example of how Threads™ might play out in areas of computing that have not traditionally been central within undergraduate computer science programs. Imagine a student, Johannes, who enters the Georgia Tech Threads™ computer science program as a freshman from Atlanta, Georgia, interested in designing computer games for handheld devices. Upon graduation, Johannes wishes to enter the large, growing computer gaming industry with unique technical skills, value-creation capabilities, and a mindset of exploration and continual learning. Via discussions with his advisor, Johannes decides to combine the MEDIA thread with the EMBODIMENT thread.

MEDIA × EMBODIMENT

The MEDIA Thread appeals to Johannes’ interest in the expressive arts (telling stories, making games, and creating emotional experiences), with courses ranging from computational graphics to Hamlet and from human perception to interactive fiction engines. Johannes and other students in this Thread learn about discrete structures, programming fundamentals and algorithms, object-oriented design and programming, operating systems fundamentals, and narrative theory and design. They will understand the computer as a medium of creative expression, be skilled in design, and experienced in modeling the structure of media (e.g., music or graphics) using dynamic data structures. Furthermore, they will be able to design and implement architectures controlling the interface between hardware and software in media devices, build discrete element simulations, and describe the impact of presentation and user interaction on exploration.

The EMBODIMENT Thread appeals to Johannes’ interest in placing intelligence in physical objects like cell phones, PDAs, and future handheld devices, with courses ranging from computational sensors to dealing with noisy data and from real-time operating systems to mobile power

¹ Scott Carlson, “Wanted: Female Computer-Science Students,” *The Chronicle of Higher Education*, v. 52, 13 January 2006.

issues and computational autonomy. Johannes and other students in this Thread learn about programming and design, numerical methods, linear algebra, dynamics and control, and circuit and sensor design. Upon graduation, they will be skilled in power management, concurrent and distributed programming, planning and reactive control, synthesizing sensors, and navigation. They will be able to design programs for resource-limited devices, implement motion planning algorithms, and design and implement a simple control architecture.

Similar to the way Johannes weaves two Threads together to form a synergistic combination, “Companies [in the computer game industry] resist segregating the disciplines of art, programming, math, and cognitive psychology and instead look for those who can piece together patches of many disciplines and weave them together into a larger tapestry.” (Pink, p. 187.) The Threads™ program will allow Johannes to be well-prepared for the current global environment in which “both the maturation of games and the offshoring of routine programming work to Asia are changing the emphasis of the gaming profession. As one gaming columnist writes:¹ ‘Changes in the way games are built indicate less of a future demand for coders, but more of a demand for artists, producers, story tellers and designers.’” (Pink, p. 187.)

Upon graduation, Johannes will not only understand *high tech* but will also understand *high concept*, a key attribute for success in the increasingly competitive economy according to Pink. High concept “involves the ability to create artistic and emotional beauty, to detect patterns and opportunities, to craft a satisfying narrative, and to combine seemingly unrelated ideas into a novel invention.” (Pink, pp. 51-52.) As such, this ability is honed via the MEDIA Thread. Combined with the EMBODIMENT Thread, Johannes will develop the symphonic-thinking needed for success as a computer science graduate.

Johannes decides to first explore the MASTER PRACTITIONER Role. In his junior year, he takes the new Real-World Lab course, in which students form development teams and solve real problems for real customers. Also in his junior year, he begins examining courses in the College of Management that might fit with his long-term goals of becoming an ENTREPRENEUR and starting his own business. During the summer between his junior and senior years, Johannes secures a paid internship at Nokia’s headquarters in Finland, enabling him to immerse himself in an organization developing the kinds of future handheld devices for which he wishes to design games. This international internship provides Johannes with first-hand experience working in a global corporation and with a glimpse into Nokia’s emerging outsourcing and offshoring strategy with businesses in countries such as India and Taiwan.

Near the end of his summer internship, Johannes attends a lecture on the great Finnish composer Jean Sibelius. He learns about a famous 1907 conversation between Sibelius and the Bohemian composer Gustav Mahler in which the two exchanged their views of the symphony. Sibelius told Mahler that symphony was about the “profound logic creating a connection between all motifs.”² While developing symphonic-thinking in real-time, Johannes connects Sibelius’s

1 Tom Loftus, “Gaming Tries to Shed Boys’ Club Image,” *MSNBC.com*, 17 June 2004.

2 The Symphony – An Interactive Guide: Jean Sibelius: <http://library.thinkquest.org/22673/sibelius.html>.

view of the symphony with the value of the College of Computing at Georgia Tech Threads™ program. Threads™ promises to develop a profound change in undergraduate computing graduates by giving them expertise in multiple areas and providing them the capabilities to create innovative connections between these areas – not unlike creating connections between all the motifs in a symphony.

2.5. Threads™ background

Across the country, top computing colleges and departments and national computing organizations have recognized that undergraduate computer science curricula have become ossified, too inflexible to meet the needs of students or the requirements for individual competitiveness. These curricula have become very good at producing a single kind of graduate – inflexible, inadaptible graduates, far from the symphonic-thinking graduates that will be leaders in the future of computing. Furthermore, current arrays of computer science programs, and their lack of creativity, create serious challenges for increasing diversity among computing students and graduates (e.g., attracting and retaining women and other underrepresented minorities). The breadth of computing and computer science has not been successfully tapped in the design of curricula. In some cases, this has led to the majority of students thinking of themselves as preparing to be professional software developers, but there are so many other possibilities. Furthermore, program enrollments are declining, and there is the perception that traditional computer science graduates are being marginalized.

In response to the disconnect between undergraduate computing programs and the new, flat world, many leading universities around the country are reviewing their computer science curricula. Many programs are approaching the problem by trying to create new computer science core curricula, and others are searching for other disciplines “to hitch their wagons to” – the concept of *computer science plus x*. In the case of the former, example new components of computer science core curricula include: international studies, core courses to broaden students’ experiences and expand their horizons, a reduction in requirements, an expanded range of capstone experiences, and interdisciplinary studies.⁴ While these changes may represent a first step, they are far from revolutionizing the way undergraduate computer science is taught and, more importantly, they will not dramatically increase the value of undergraduate computing graduates. In the case of the latter, the implementation of CS + x presents a sustainability challenge for universities and poses the serious risk of leading to short-term quality compromises and the pursuit of short-term “trendy ideas.”

While there has been investment in computing-related education innovation at the graduate level for many years,² there have been no major innovations to undergraduate computer science curricula and programs. Emerging from a national discourse surrounding the need for innovation

¹ Larry Finkelstein, “Restructuring Academic Programs for a Global Based Knowledge Economy,” presentation at 2004 CRA Conference at Snowbird, July 2004.

² In particular, the IGERT (Integrative Graduate Education and Research Traineeship) program sponsored by the National Science Foundation (<http://www.nsf.gov/crssprgm/igert/intro.jsp>).

in undergraduate computing education, The College of Computing at Georgia Tech is on the forefront via the development of Threads™, its implementation set to begin in Fall 2006.

Threads™ represents a new way of organizing undergraduate education. The current curriculum in the College of Computing at Georgia Tech suffers from a perceived lack of both flexibility and transparency. Students and faculty alike complain of a “one size fits all” computing degree that produces fairly narrow students. In a college as diverse and broad as the College of Computing at Georgia Tech, this makes little sense. Insofar as there is flexibility, it is difficult to communicate to students. While there is a notion of specializations, they are often first considered only at the senior level, and as such, students do not have an occasion to prepare themselves for succeeding at those specializations by pursuing, for example, skills taught in classes outside of the College of Computing.

Threads™ represents a tremendous departure from how people think about curricula in computer science and computing – a departure from a vertically-oriented curriculum whose goal is the creation of students with a fixed set of skills and knowledge. Threads™ does away with the idea of a monolithic core curriculum plus a pool of various electives (for which there often exists no clear “selection algorithm” for students to use in making choices). Computer science (and more broadly, computing) as a discipline is an increasingly broad spectrum. Threads™ gives students the power to select where they want to be in this spectrum and the outlook needed to see how their individual curricula fits into the “big picture” and contributes to students’ lifelong learning and competitiveness in an interconnected, global environment.

The College of Computing at Georgia Tech

The College of Computing at Georgia Tech houses one of the largest computer science programs in the country. The undergraduate major in computer science is the largest and one of the broadest at Georgia Tech. Over 1,200 undergraduates are currently enrolled, with over 200 students pursuing Master’s degrees and nearly 300 students pursuing Ph.D.s in the College. The College houses 68 academic faculty and 39 research faculty. As one of the few institutions with a college-level computing program, the focus at Georgia Tech is on computing, not just the discipline of computer science. The College’s educational and research activities collaborate with many other disciplines in order to push the frontiers of computing throughout the Institute.

The history of the College of Computing began in 1963 with the establishment of the School of Information, giving Georgia Tech the distinction of having one of the first computer science programs in the country. The College of Computing was formally recognized with College status in 1990 and given a charge to promote interdisciplinary research and education.

Appendix A. Descriptions of Individual Threads

Eight dynamic Threads have been defined, enabling 28 possible combinations of two Threads.

Computational Modeling

The COMPUTATIONAL MODELING thread is where computing meets and describes the world. COMPUTATIONAL MODELING prepares students with the technical knowledge and skills necessary for expressing, specifying, understanding, creating, and exploiting computational models that represent cognitive and physical processes. It prepares students for fields as diverse as artificial intelligence, machine learning, perception, cognitive science, and graphics. The student who pursues COMPUTATIONAL MODELING can combine it with EMBODIMENT to become a roboticist, or with PEOPLE to build adaptive interfaces, or with MEDIA to prepare for a career in graphics, or with ...

Early Preparation

- Combinatorics
- Numerical methods
- Linear algebra
- Probability and Statistics
- Discrete structures, graph theory
- Object-oriented design and programming

Knowledge Goals

- Understanding statistical inference (e.g., building optimal models from noisy and complex data)
- Understanding computational methods for dealing with massive and high-dimensional datasets
- Facility with numerical methods (e.g., algorithms for dealing with continuous functions)
- Facility with performing massive-scale computations

Skill Outcomes

- Able to implement a variety of pattern recognition and control algorithms, and understanding the applicability of each
- Able to deal with high-bandwidth streams of data
- Able to develop autonomous systems in a variety of domains

Embodiment

The EMBODIMENT thread is where computing meets the world. It is concerned with computation that operates under possible severe physical constraints. Via EMDODIMENT one learns how to create and evaluate computational artifacts that are embedded in physical objects and interact in the physical world, typically in real time. The student who pursues EMBODIMENT can combine it with PLATFORMS to build devices that are small, power- and CPU-limited, or with COMPUTATIONAL MODELING to build autonomous robots, or with PEOPLE to study human-robot interaction, or with ...

Early Preparation

- Programming and design
- Numerical methods
- Linear algebra
- Dynamics and control
- Circuits and sensor design

Knowledge Goals

- Power management
- Concurrent and distributed programming
- Planning and reactive control
- Synthesizing sensors
- Navigation

Skill Outcomes

- Ability to design programs for resource-limited devices
- Ability to implement motion planning algorithms
- Ability to design and implement a simple control architecture

Foundations

The FOUNDATIONS thread is where computing meets itself. FOUNDATIONS teaches students the theoretical and mathematical foundations underlying a wide range of computing disciplines. The student who pursues FOUNDATIONS can combine it with EMBODIMENT in order to provide performance bounds for robotic planning algorithms, or with PLATFORMS to become a researcher in programming languages, or with ...

Early Preparation

- Discrete structures
- Proofs and proof techniques
- Basic algorithms (e.g., sorting and searching) and algorithmic techniques (e.g., greedy, divide and conquer)
- Basic computability theory
- Numeric methods (number representation, error, stability, basic numerical methods and algorithms)
- Calculus, including differential equations
- Linear algebra
- Combinatorics
- Probability and statistics

Knowledge Goals

- Complexity classes and NP-completeness
- Automata theory
- Advanced algorithms and analysis (e.g., amortized analysis, online & offline algorithms, randomized algorithms)
- Cryptography
- Parallel and distributed algorithms
- High Performance computing
- Continuous and discrete simulation (e.g., modeling paradigms, analysis, validation)
- Computational problem solving in selected application domains (e.g., physical sciences and operations research)

Skill Outcomes

- Ability to develop complex algorithms taking into account fundamental limits (e.g., computability, complexity bounds) and real-world constraints
- Ability to analyze and prove the correctness of algorithms

Information Internetworks

The INFORMATION INTERNETWORKS thread is where computing meets data. Information-centric computing prepares students for information management by helping them to capture, represent, organize, transform, manage, and present information securely and efficiently. The student who pursues INFORMATION INTERNETWORKS can combine it with COMPUTATIONAL MODELING to study text retrieval and classification, or with PEOPLE to pursue research in data visualization, or with ...

Early Preparation

- Computer architecture, hardware, and operating systems
- Discrete structures, graph theory
- Object-oriented design and programming
- Data structures and programming skills
- Computer hardware architecture
- Basic operating systems
- Basic communication networks
- Discrete structures: set theory, graph theory
- Basic algorithms (e.g., sorting, searching) and algorithmic techniques (e.g., greedy search, divide and conquer)
- Probability and statistics
- Combinatorics

Knowledge Goals

- Communications and networking architectures and protocols
- Data security and privacy
- Mobile computing
- Interoperability issues, distributed object systems and middleware
- Data modeling and conceptual models
- Relational data models, object-oriented models
- Database implementation and principles, data storage, indexing, query optimization, recovery, transactional model, concurrency control, and scalability
- Indexing, searching, and mining for Web and sensor stream data, multimedia data management, integrating heterogeneous data sources
- Data visualization principles, especially for large datasets

Skill Outcomes

- Able to implement network protocol stacks in the context of contemporary operating systems
- Able to program with sockets, threads, IPC
- Able to implement secure, reliable client/server and peer-to-peer distributed systems
- Able to develop effective, distributed applications (e.g., multimedia)
- Able to develop data models
- Able to select, design, and implement scalable and secure information management solutions
- Able to develop effective client-server database applications
- Familiar with programming languages such as Java, C++, C and principles of implementing secure stand-alone and server side applications

Intelligence

The INTELLIGENCE thread is where computing meets and models intelligence. INTELLIGENCE is concerned with computational models of intelligence from top to bottom, with an emphasis on designing and implementing artifacts that exhibit various levels of intelligence as well as understanding and modeling natural cognitive agents such as humans, ants, or bees. Students acquire the technical knowledge and skills necessary for expressing, specifying, understanding, creating, and exploiting computational models that represent cognitive processes. It prepares students for fields as diverse as artificial intelligence, machine learning, perception, and cognitive science, as well as for fields that benefit from applications of techniques from those fields. The student who pursues INTELLIGENCE can combine it with EMBODIMENT to become a roboticist, or PEOPLE to build adaptive interfaces, or with MEDIA to build smart, adaptive entertainments, or with...

Early Preparation

- Combinatorics
- Numerical methods
- Linear algebra
- Probability and statistics information theory
- Discrete structures, graph theory
- Object-oriented design and programming

Knowledge Goals

- Reasoning with uncertainty
- Reasoning on action and change
- Heuristic methods for solving problem that are difficult or impractical to solve with other methods
- Techniques for handling high-dimensional spaces
- Modeling static vs. dynamic worlds

Skill Outcomes

- Able to implement a variety of pattern recognition and control algorithms, and understanding the applicability of each
- Able to build fast approximation algorithms for dealing with high-bandwidth streams of data
- Able to develop autonomous systems in a variety of domains

Media

The MEDIA thread is where computing meets design. MEDIA prepares students by helping them to understand the technical and computational capabilities of systems in order to exploit their abilities to provide creative outlets. The student who pursues MEDIA can combine it with MODELING to study animation, or with INFORMATION INTERNETWORKS to high performance database systems, or with PEOPLE to explore visualization of high-bandwidth data streaming, or with...

Early Preparation

- Discrete Structures
- Programming fundamentals and algorithms
- Object-oriented design and programming
- Operating Systems fundamentals
- Narrative theory and design

Knowledge Goals

- Understanding the computer as a medium of creative expression
- Design
- Modeling the structure of media (e.g., music or graphics) using dynamic data structures

Skill Outcomes

- Able to design and implement architectures controlling the interface between hardware and software in media devices
- Able to build discrete element simulations
- Able to describe the impact of presentation and user interaction on exploration

People

The PEOPLE thread is where computing meets users. PEOPLE prepares students by helping them to understand the theoretical and computational foundations for designing, building, and evaluating systems that treat the human as a central component. The student who pursues PEOPLE can combine it with EMBODIMENT to study human-robot interaction, or INFORMATION INTERNETWORKS to pursue research in data visualization, or with COMPUTATIONAL MODELING for learning sciences and technology, or with PLATFORMS to explore ubiquitous computing, or with...

Early Preparation

- Discrete Structures
- Programming fundamentals and algorithms
- Object-oriented design and programming
- Concurrent processing and event-driven programming
- Experience building simple GUIs

Knowledge Goals

- Understanding human behavior with interactive objects
- Knowing how to develop and evaluate interactive software using a human-centered approach
- General knowledge of HCI design issues with multiple types of interactive software.
- Achieve a depth of understanding of some aspect of people: perceptual, social, motor, cognitive, and so on

Skill Outcomes

- Able to apply HCI techniques to identify usability problems and gather design requirements
- Able to build working systems to address those problems
- Able to apply qualitative and quantitative techniques to evaluate the success of those systems

Platforms

The PLATFORMS thread is where many of the practical skills of computing are learned. Like the FOUNDATIONS thread, PLATFORMS lies at the center of computing. It prepares students to create and evaluate computer architectures, systems and languages across a variety of paradigms and approaches. The student who pursues PLATFORMS can combine it with FOUNDATIONS to study distributed high performance computing algorithms, or with INFORMATION INTERNETWORKS to study real-time data retrieval systems, or with PEOPLE to pursue research in developing programming environments, or with...

Early Preparation

- Programming and design
- Basic Communication networks
- Discrete structures: set theory, graph theory
- Algorithms and algorithmic techniques
- Probability and statistics

Knowledge Goals

- Operating systems
- Concurrent and distributed programming
- Architecture performance measurement and evaluation
- Architecture techniques (e.g., pipelining, memory hierarchies)
- Parsing techniques
- Typing and semantic analysis
- Language paradigms and design

Skill Outcomes

- Compiler construction
- Parser generation
- Familiarity with current computer architectures and their features
- Evaluating computer performance
- Programming in different paradigms and multiple languages

Appendix B. Descriptions of Individual Roles

Four Roles have been defined and additional possibilities for students exist, including the ability to explore more than one Role.

Master Practitioner

Expert programmer who possesses the technical skill and experiences to thoroughly design, construct, and validate computer-based systems either alone or as a part of a large team. The MASTER PRACTITIONER is the Programmer writ large. The master practitioner is not just a hacker or code monkey, but a person who can apply careful abstraction to any problem and implement a well-designed solution in one of several programming languages and styles. The MASTER PRACTITIONER is interested in the exercise and mastery of technical skill and is likely to be an employee of a corporation.

To be a MASTER PRACTITIONER, one must demonstrate advanced competence in a number of programming languages of a variety of types to a level comparable to industry certification standards. The MASTER PRACTITIONER will eventually be happy working at the micro-code level, or developing for desktop machines or cell phones.

There are several options for demonstrating this level of competence. These include the new Real-World Lab course (CS 4981), allowing formation of a development team that solves real problem for a real customer, and the new Architecture Studio course (CS 4991), allowing intense involvement in practical techniques such as pair programming.

Entrepreneur

Creator and leader of new enterprises that bring technology to the public at large, specifically in the form of new products and services. The model of an ENTREPRENEUR is the founder of a start-up company that provides products and services that impact or transform society.

Because such companies require a technological vision as well as a marketing one, the ENTREPRENEUR must be deeply knowledgeable about and invested in the technology that forms the core of the enterprise as well as passionate about making that enterprise economically viable. An ENTREPRENEUR may also find a home as an employee of a corporation that rewards innovation.

The interested student should participate in a set of entrepreneurial activities and perhaps take one or more classes in the College of Management and perhaps participate in the new Undergraduate Business Opportunities in Computing (UBOC) program for course credit, a program that facilitates the formation of interdisciplinary College of Computing and College of Management teams under the mentorship of local business leaders. Participants learn to develop business plans for computing enterprises, and demonstrate their ideas and in a competition for capital for a start-up. Teams are typically interdisciplinary and bring together the pieces necessary to bridge technology with the marketplace.

Innovator

Discoverer of new knowledge and constructor of ground-breaking solutions to problems. The model of an INNOVATOR is the academic or industrial research scientist making discoveries and inventions. These discoveries will eventually make a difference in society but result from investigations that are not guaranteed to yield practical results.

Students who are considering eventually pursuing a Ph.D. should consider exploring research opportunities in the College of Computing, but the INNOVATOR Role is also appropriate to students who do not expect to continue their academic careers beyond a Bachelor's degree. The Georgia Tech Research Institute, for example, hires research-oriented undergraduates, as do many other technologically innovative companies across the nation.

To be considered an INNOVATOR, a student should participate in one or more research experiences during their tenure at the College of Computing. There are several opportunities, including CS 3901 (Research Project) and the Undergraduate Research Opportunities in Computing (UROC) program.

Communicator

Individual capable of communicating technical information to the technologist and the layperson alike. The COMMUNICATOR is technically skilled and informed and works hard developing the communication skills necessary to effectively share their knowledge with interested but less expert audiences.

The model of a COMMUNICATOR is the excellent and engaging lecturer, or the author of a popular but technically deep text, such as the O'Reilly "In A Nutshell" series. The COMMUNICATOR is motivated primarily by the desire to clarify difficult technical material and convey it in a way that others can understand and apply. The COMMUNICATOR might become a corporate trainer/consultant, teaching faculty, a freelance writer, or perhaps a researcher in computing education.

The interested student should demonstrate effective communication skills both orally and in writing. Formal experiences might include a stint as a teaching assistant in one or more courses, eventually involving direct communication with students in a laboratory or lecture setting. Furthermore, the student might undertake an internship with a campus computing group (e.g., the Office of Information Technology at Georgia Tech or the Computing and Networking Services group within the College of Computing) writing effective technical documentation or user tutorials.

