Use of Failure in IS Development Statistics: Lessons for IS Curriculum Design

Herbert H. Longenecker, Jr.
longeneckerb@gmail.com
University of South Alabama
Mobile, AL 36688

Jeffry Babb
jbabb@wtamu.edu
West Texas A&M University
Canyon, TX 79116

Leslie Waguespack
lwaguespack@bentley.edu
Bentley University
Waltham, MA 02452

William Tastle
tastle@ithaca.edu
Ithaca College
Ithaca, NY 14850

Jeff Landry
University of South Alabama
jlandry@usouthal.edu
University of South Alabama
Mobile, AL 36688

Abstract

The evolution of computing education reflects the history of the professional practice of computing. Keeping computing education current has been a major challenge due to the explosive advances in technologies. Academic programs in Information Systems, a long-standing computing discipline, develop and refine the theory and practice of computing appropriate to professional practice. As the computing professions advance, so too does our conceptualization and design of curricula in information systems. Subsequently, our organizing bodies (i.e. DPMA, AITP, AIS, ACM, etc.) coordinate and cooperate in the development of curricular guidelines and models. These models serve to establish shared core and methodology among practitioners and educators. This paper presents the case that there are lessons in the history of the computing fields – particularly among the system development failures – that can inform the design of curricula aimed at preparing computing students for professional practice. Given the repetitive nature of many of these failures, we posit that failures can shed light into the "dark places," and, with care, illuminate the essential nature of the information systems discipline and the body of knowledge and skill sets essential to our educational task.

Keywords: Computer Information Systems (CIS), Building CIS Programs, CIS Curricula and Specialties, Body of Knowledge

1. INTRODUCTION

In the 1960's, as the computing sciences emerged, it became obvious that educated professionals were needed. This led to the development of university programs of study with a naturel division into the (1) development of both the hardware and software of the machines and (2) business and organizational applications. Over time, Academic curricula developed for each area. The hardware characteristic at the time including small memories, limited peripheral storage, and severely restricted input and output was a necessary curricular consideration of all academic programs. From these earliest beginnings, a consistent sentiment is well expressed by Gordon Davis**,** "Information systems is about the development and deployment of systems." (Davis, 1994) While the words development and deployment have evolved, the concept remains similar. As educators in the field of Information Systems, we are reminded that "computer" is not a dirty word to be avoided; it is the wellspring that binds the computing disciplines and allows for Information Systems to ground itself. To wit, some educators and programs have taken to claiming this heritage by prefixing "computer" to the program. We investigate the origins of this sentiment and explore whether Gordon Davis' sentiment remains relevant.

This paper outlines an argument that as computing education has evolved, propelled by computing's successes and its transformative power, there have been numerous and significant failures. While the litany of these failures is entrenched in our canon, we should recognize these failures – of specification, design, development, and implementation – as evidence of blind spots: core deficiencies in the professional practice of computing that require more curricular attention – particularly in the academic discipline known generally as "information systems."

Among the marvels of human history is how computing power has grown considerably and consistently since the inception of the various computing disciplines. In the nascency of computing disciplines, computers were programmed using primarily the FORTRAN and COBOL languages. As tools like these were widely used, curricula specifications in computing, such as ACM '68 and ACM '71, accounted for curricular formulations that reflected the needs of industry. At the time, in the business domain, the computing function was mainly accounting and reporting functions (to the CFO and upper level management). As academic programs emerged to support the computing needs of government and businesses, a graduate's professional track was typically to first be a programmer, and then a progression to analysts and beyond (eventually these functions required management). This progression was consistent such that there was little diversity of expectations; most everyone went through these stages, even up to the mid 1990's.

The Rise of Computing in Business

From its beginnings in the 1960s, computing increasingly factored into daily life, and subsequently into culture. For instance, the January 1983 issue of TIME magazine had the IBM PC as the "man of the year." Given the impact of shrinking computing architectures such that "everyday" people could compute in their own homes, public awareness of computing, and the impacts it was having on daily life, had grown tremendously. Subsequently, enrollments in university programs in computing-related disciplines became very large. With wider exposure, the public quickly realized that the PC was far from trivial to learn and use; to the contrary, many realized that attempting to get these machines to do anything often resulted in adversity and outright failure.

It seems that computing finds new traction in the public imagination every 5 years or so (the PC, Software, The Internet, The World Wide Web, Web 2.0, Mobile, etc.). In these cases, a gold rush mentality develops, and enrollments in computing programs spike, and then the realities of actually working with these new technologies set in, and, enrollments return to their previous levels. While computing, and the academic programs that have arisen to develop professionals in computing, have matured and diversified, the core knowledge and fundamental skill sets remain largely unchanged. During the earliest expansions and contractions of student interest in academic programs leading to a career in computing, computing education retained its essential focus: the application of problem solving and logic through the medium of the computer to produce useful business tools, computing artifacts. The surge of student interest mirrored the rapid realization in most organizations of the power and potential of computing to exploit information for business success.

In the 1970s and 80s, the computer center director and his million-dollar machine owned the landscape as long as the payroll was produced on time. It was rarely possible to win in an outright slugfest against this power. Upon the entry and ascendancy of the PC, a counter to the power of central computing was the fact that the many varieties of the PC were both affordable and powerful. What ushered in has become known as the end-user computing era – functional units had computing power right in their areas, and would call upon computing professionals to develop applications and systems around these machines. As they became inter-networked, this reallocation of power became even more pervasive. Even in cases where the director of computing controlled institutional purchasing policy, the lure of opportunity and proliferation of the PC was hard to restrain. The imagination and creativity of many functional managers were fired by the PC's potential: hundreds if not thousands of new types of applications could be developed. The earliest movements to decentralize computing brought on new demand and interest; computing grew more influential in the business culture and the collective consciousness.

Computers Get Smaller, Cheaper and Omnipresent

By the 1980s, large firms were no longer the only organizations depending on computing. Both the cost and complexity placed computing within the reach of medium and many smaller organizations. This trend continues today. Moreover, the shift in the nature of business and organizations reshaped through computing empowers many small businesses to flourish. Consider the result of a 2015 SBE Council Report:

"In 2011, according to US Census Bureau data, there were 5.68 million employer firms in the United States. Firms with fewer than 500 workers accounted for 99.7% of those businesses, and businesses with less than 20 workers made up 89.8 %. In the number of non-employer firms in 2012 – there were 22.7 million in 2012 –the share of US businesses with less than 500 workers increased to 99.9%, and the firms with less than 20 workers increased to 98% (SBE Council, 2015). Clearly, there was considerable growth in the base and with it, small to medium sized companies brought computer technology into their operations, utilizing them in similar ways for similar purposes.

Hence, hindsight reveals a number of very significant factors that developed during the mid-1980s and 1990s that produced the basis for a remarkable growth-pattern of business computing during the latter 1990's and beyond. Among these factors were: growth of the electronics industry in building new "chips", cheaper disk storage, and un-paralleled communications capability. What also emerged were patterns of expansion and contraction as the tools of computing, and creative (and profitable) applications thereof, advanced. However, what seems to be the case in most of the boom/bubbles created around computing is that the essence of the artifact had not changed fundamentally, whereas uses did as computing expanded, improved, and was ever more available.

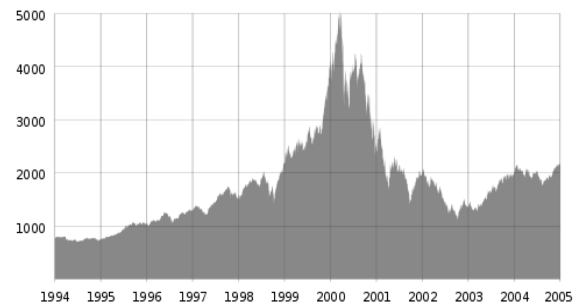


Figure 1 - The NASDAQ Composite index spiked in the late 90s and then fell sharply as a result of the dot-com bubble. https://www.google.com/webhp?sourceid=chrome-instant&ion=1&espv=2&ie=UTF-8#q=dot-com%20bubble

Widespread Proliferation

Computing witnessed another bubble around 2000, not unlike the case with the PC: the dot-com bubble. While the bubble itself did not take place until just after 2000 (WiseGEEK, 2015) several technological advances led the business community to invest very large sums of money in new internet dependent start-ups. More than 60% of these organizations failed. What both the PC and Dotcom bubbles reveal is how little is known about what computing is and is not. Fad-like and fleeting, extant ideas were repackaged in new buzzwords and theories where, in fact, the essentials of computing were not intrinsically different. Certainly, advances in computing hardware are there, as is the proliferation of content via the Internet and its TCP/IP architecture, but these are evolutionarily advanced upon us which, as they proliferated, of which industrious and entrepreneurial individuals took advantage. There can be little doubt that the significant effort released by the CERN group (Berners-Lee, 1991) as reflected by Lee's discovery of the WWW. In general, there is lack of public awareness and understanding of computing which has also given rise to other peripheral phenomenon like information security and privacy, specialization in network and systems administration, and what is commonly referred to as "IT." However, at its core, the actions and goals driving end-user computing as well as their development, and the "dot-com" era, are still rooted in the activities described by Gordon Davis – "…the development and deployment of systems." With this widespread proliferation of massively inter-connected computing, society grew so enamored with what computers "can do" they began to ignore the fact that people need to know "how computers do it;" this leads to the illusion that knowing "how to use" and "how to develop a computer system" are somehow isomorphic.

Rise and Fall

The context of the computing disciplines, as they exist today, is best understood if the conditions surrounding the dot-com bubble are understood. Rather than focus on economic theory surrounding the propensity for "bubbles" to form in concert with the ebb and flow of market forces, it is best to discuss the evolution of computing such that the conditions that fostered the dot-com bubble are understood. To understand them is to realize that the fundamentals of computing continue to move at an evolutionary pace, interspersed with remarkable advances in the power and availability of computing.

The brief (and interpretive) recounting of computing history provided thus far reveals patterns in the computing disciplines that reflect the nature of computing as mechanized information processing – a nature that does not change with the advance of the technologies that mechanize it. We do not seek to identify these fundamentals in an exhaustive and conclusive manner; we only seek to illustrate that our own history can partially refocus who and what we are, as educators and practitioners of the computing discipline generally known as information systems. Our position taken here is that to discount or ignore those fundamentals is folly that cannot be overcome by technological advances.

2. FACTORS CONTRIBUTING TO THE ONSET OF THE "DOT-COM" ERA

Millions of Potential Computers

The 1983 TIME article regarding PC, if nothing else, put people and organizations on notice there would be a lot of computing capacity for all who were desirous of it. As the hardware advanced, multi-user operating systems also advanced supporting remote computer access and otherwise inter-connecting computers. The infrastructure enabling the inter-computer/inter-user data connections were mechanisms first used in the 1960s and, via evolutions in computing, are still the foundation of what is in use today. However, there is little doubt (historically) that networking is the technology that really transformed an otherwise home computing environment into a connected purposeful structure (Berners-Lee, 1991). Networking extended systems and their information to wider contexts and more people. Thus, the impact of computing has been profoundly felt amongst people and that awareness of the "what" was possible with computing also increased.

3. FACTORS WHICH MAY HAVE ACCELERATED THE COLLAPSE OF THE DOT-COM BUSINESS

The WWW is both a plus and minus to business in the 1995-1998 era. On one hand, since the hypertext protocol would run on the average office PC, any PC with the right hardware (A network interface card to use the Ethernet standard) and software (a web browser), could connect to any other host (PC, server, or otherwise) which was discoverable using a combination of TCP/IP and DNS. Since a "server" could be located anywhere, then the PC could serve as an input device for a program running on the server in a multi-user environment. A catalog order-entry inventory system would be a possible system to utilize such an architecture — Amazon comes to mind as a company that capitalized early on the mechanics that eventually promulgated most of the successful "dot-com" ventures. Ideas of this magnitude are very attractive. However, it is important to revisit the following proposition: as computing grows and continues to change the human (and natural) environment, many humans remain unaware or ignorant of the fundamental nature of what computing is (and is not). As is the case with many human endeavors, while computing is behind a plethora of successes, computing also has much to teach about failure. It is not that the hardware is principally flawed (although possible, these errors are usually ironed out), it is rather the "soft systems" that are often causal in computing failure. To wit, an entire sub-discipline of computing is alive and well, and will likely remain so into the distant future, because of human failure to understand computing: IT support. Adages like PEBCAK (Problem Exists Between the Chair and Keyboard) are humorous responses to real problems – a fundamental failure to become familiar with what is computing.

As we ruminate on the dot-com phenomenon in order to understand the progression of computing, it seems that business owners and investors had not read the Standish Group's

CHAOS Report (1995, 2001) which cited significant failure statistics for any IT/Software project. Indeed, they predicted the failure statistics for new software to be at about 80% - which means 4 out of 5 software projects (of which nearly all significant "IT Revolutions" have centered on) will fail. The translation of these results to a business was that business planning and requirements were poorly conducted if at all. During the late 1990's it was also uncertain if requirements could be translated into good code. Unfortunately, investors lost track of the facts and in the dot-com area spent all of their cash on the anticipated success of these ideas. As all of investors' funds were consumed, the "dot.bust" era resulted in the loss of an enormous of cash. Why? Again, we proffer a simple proposition: the average individual is largely ignorant as to what computing is (and is not), and what it takes to generate a good solution. Moreover, the Standish Group's CHAOS report presents the case that this ignorance is often willful and deliberate. While most would agree that the age of enlightenment, science, and significant progress brought about by technologies has negated a tendency to invest in magic, and yet evidence continues to hold that computing is largely treated as though it were magic.

4. LESSONS FROM FAILURE: CURRICULA NECESSARY FOR DEVELOPMENT

The motivation for this section is based on a Failure Analysis of Information Systems of the Standish Group (1995, 2001). While we are sympathetic to software development firms as well as the academics who trained the workers who wrote the defective software, we must highlight these failures as a reminder of our raison d'être. As an illustration, we reflect on two such information systems failures:

a. The Affordable Care Act website launch of 2013
b. Denver Airport Baggage handling system failures of 1995

Both of these systems fit the general observation of Standish Group failures:

a. Totally inadequate business planning and requirements development
b. Poor project management
c. Enormous cost overruns
d. No concept of task
e. Many intended specs abandoned
f. Embarrassed leaders who tended to lie about results and cover them up

In the case of the Affordable Care Act website launch, it turns out that 55 contractors were involved in developing and deploying this failed system launch for a considerable cost of $400 million. It is somewhat exacerbating to learn that the company that fixed it was a small agile team in which the cost of repair was only $4 million. Given the controversy surrounding the legislation itself, it would seem that the failures of the ACA website would have been avoided, but many of the root causes lie within the ACA project itself and have become well-known through studies and reports on software systems development over the years. Again, one must question the levels of ignorance that persist in projects that seek to harness the power of computing.

The exposition of aspects of the history of computing, and the degree to which we can relate this history to the computing disciplines, has been presented as a means of understanding the task before those who consider curriculum design for computing disciplines such as information systems. Among the issues faced is occurrence of the boom-bust cycle as the evolutionary advances of computing build to the point that revolutionary uses are manifested in a rapid manner in markets. Thus, the demand for professionals proficient in computing rapidly expand during these times. Throughout the short history of the computing professions, both market and management failures have not lead to consistency in how computing is situated within society. What professional imperatives exist, if any, when we continue to grapple with the failure statistics presented by the Standish Group. When these matters are considered from an educator's perspective, one must look to what can and cannot be done in academia. We develop curricula to serve as a guide that gives cohesion, focus, and definition. What responsibility do we have in academia to share in the blame for these failures? Are we lacking in project management? Surely, and perhaps certainly, in the last 20 years it is likely that these failed projects have involved well-qualified, certified, and perhaps even experienced project managers. Even if our computing curricula, particularly in the case of information systems, were to be turned over to being largely about IT project management, we contend again, propositionally, that public willful ignorance of computing would doom these individuals. Verily, through the experiences of some of the authors of this paper, many undertaking training in the various forms of IT and software project management regimens and paradigms do not really understand what is computing. For some programs, appending "computer" prior to information systems is not a glib marketing strategy; it is a reminder of the primacy of understanding computing. As an extension, we reflect on Gordon Davis' words - Information systems is about the development and deployment of systems – we realize that these "systems" rely on a non-trivial understanding of what is computing and what is required to make artifacts which reliably serve this purpose. Software engineering has arisen to take this mantle, and challenges information systems educators to decide who they are, what they represent, and how this shall be delivered as a curriculum to produce graduates who are useful in the computing discipline.

Orienting Curriculum Design

Persistent failures in information systems implementation can also be seen as an opportunity. A possible "prescription of a cure" to inoculate computing professionals from the "folly" of technology solutions is to design our curriculum such that computing professionals understand the limits of the mechanization and the critical essence of understanding "information and organizational processes" in computer system asset value. Put metaphorically, "building safe and functional furniture rests more in the skill and knowledge of the woodworker's tools then it does in the intended purpose for the table or chair!" Our task in the development of information systems curricula is to provide this assurance. Moreover, we should ensure that society is concerned for and supportive of an appropriate computing curricula. Our curricula should underscore how and why professional competency is relevant.

We suggest that striving for excellence in information systems must involve great education for students and professionals. We further suggest generation of great systems must revolve on teams of IS professionals who are skilled in the applications as stated herein.

A. For great systems:

1) A set of clear exit objectives toward which curriculum productions are aimed. These objectives, or outcomes, will describe the behavior of IS professionals in developing excellent information systems;
2) A body of knowledge representing all the necessary ingredients of curricular skill specifications. These skills must describe behaviors which when combined appropriately and executed will reliably produce behaviors associated with the exit objectives;
3) A clear pathway which may be constructed connecting the body of knowledge and derived productions to the exit objectives wherein sub-exit-knowledge products which provide successively, a linked network mapping the body of knowledge to the exit objectives.

B. For systems built without excellence—potential failure:

Lack of excellence in IS development—i.e. failure—is:

1) Expensive to organizations as well as individual investors;
2) Represents an unwillingness to search development methodologies for steps that might lead to success, including project management.

Figure 2 moved to the appendix (Curriculum Models: A) is a general model, B) is the model of IS'97 model, and C) is the Model of a to-be-proposed 2016 CIS model curriculum.)

We are impressed with the vastness of the Standish group 44,000+ samples which represent a significant degree of failure. Likewise, their focus on what makes a successful system gives great credibility to their recommendations. Appendix 1 is a set of abstractions from our analysis of the Standish Groups effort. We argue that these statements represent a positive set of elements that might well represent the most significant requirements for a body of knowledge (Figure 2C): If you followed these constructs we are determined you would have the capacity of generating a great system.

Appendix 2 is the detailed body of knowledge (see also Figure 2C) recently determined by survey of a group of professionals (Longenecker, 2015). Within this document is Appendix 3, which is in an abstraction of Appendix 2. For a challenge, the fidelity of the mapping of appendix 3 to Appendix 1 was studied. For each element of Appendix 3 we then asked if the individual element provide support for one or more elements of Appendix 1. The procedure was repeated for the abstracted body of knowledge, Appendix 3. For both appendix 2 and 3 we found a very good match of new proposed body of knowledge (Appendix 2 and 3) with the Standish group recommendations (Appendix 1).

Appendix 4A contains the body of knowledge for IS'97. The elements of Appendix 4A map reasonably well to the exit skills shown in Appendix 4B determined by the IS'97 task force.

Information Systems curriculum development is supported by previous model curricula. Earlier models contain work that has been expanded over the years as the field has grown. The early documents initiated and paved the way for current work. These documents include:

- ACM 71 – Teichroew, 1971; Nunamaker et al 1982
- DPMA 80
- DPMA 86
- AITP 90 – Longenecker et al, 1991
- AITP 95 – Couger et al, 1995; Gorgone et al, 1994; Longenecker et al, 1995
- AITP 97 –
- AITP/ACM/AIS 2002 – Gorgone, et al; (Landry et al, 2000)
- AIS/ACM 2010 – Topi, et al, 2009, 2010

Documents most pertinent to students educated during the dot-com era were those of IS'95 and IS'97. The Body of Knowledge for these documents were identical and are reprinted as Appendix 4. These elements although they were widely reviewed and accepted, do not map very well to appendix 1. Therefore, and unfortunately, the exit objectives of Appendix 4B do not map well to the currently identified elements of Appendix 1. The meaning of this incomplete mapping is that information systems curriculum constructed (Figure 2B) based on these objectives may well be deficient.

5. CONCLUSIONS

We have presented a case that, by understanding some key transitions in the history of computing, coupled with the great promises and failures of computing technologies over the years, we may better know why the information systems discipline may exist and what purpose it may serve. Many could bear testimony that information systems, as an academic discipline, has benefitted from the many "rising tides" across the last 60-odd years of computing as an academic pursuit. However, we have also, directly or indirectly, been party to the failures of computing. This is an odd sentiment to express as computing, in and of itself, is mathematically sound such that "computing" isn't failing but rather, it is our utilization of computing to deliver information systems that are useful, reliable, and robust. The Standish Reports have been examined to illustrate the point that, just as the fundamentals of computing are consistent as they evolve (thus far), so too are the root causes of failure. When an industry produced "failed" or "distressed" projects 4 out of 5 times, there certainly is enough blame to go around. In an attempt to connect the various high-water marks in our history, we propose that the point is past where collective willful ignorance can stand much longer. In the evolutionary process of defining and refining a model curriculum for the discipline, it would seem that a centerpiece of renewed efforts to define and refine should take the Standish Group (and that of others) findings and utilize these persistent problems as a base. While the marketplace will always provide those who can connect the promises of computing technologies to consumers, we must not wait for rising tides to define a forced discipline upon us. Perhaps starting with persistent failure is a good way to define our purpose and worth.

In conclusion, the fidelity of the mappings between appendences 2 and 3 with appendix 1 indicate if we build a curriculum based on our body of knowledge, we may be able to avoid the issues that plagued traditional software developers. We feel that this method of triangulation will produce the most useful results.

6. REFERENCES

ACM Curriculum Committee on Computer Science 1968. Curriculum 68: Recommendations for the Undergraduate Program in Computer Science. Communications of the ACM, 11:3, March 1968, pp. 151-197.

Berners-Lee, T (1991). "The birth of the world wide web," retrieved at http://timeline.web.cern.ch/timelines/The-birth-of-the-World-Wide-Web

Couger, J. D., Davis, G.B., Feinstein, D.L., Gorgone, J.T. and Longenecker, H.E. (1997). IS'.97: Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, Data Base, Vol. 26 No. 1, pp. I-94.

Couger, J. D., Davis, G. B., Dologite, D. G., Feinstein, D. L., Gorgone, J. T., Jenkins, M., Kasper, G. M. Little, J. C., Longenecker, H. E. Jr., and Valachic, J. S. (1995). IS'95: Guideline for Undergraduate IS Curriculum, MIS Quarterly (19:3), 1995, pp. 341-360.

Davis, G., J. T. Gorgone, J. D. Couger, D. L. Feinstein, and H. E. Longenecker. (1997). IS'97: Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems. ACM SIGMIS Database, 28(1).

Davis, G.B., Couger, J. D., Feinstein, D.L., Gorgone, J.T. and Longenecker, H.E. "IS '97 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems," ACM, New York, NY and AITP (formerly DPMA), Park Ridge, IL, 1997.

Dot-com Bubble, David Schepp (2001). 'Warren Buffet: I told you so', BBC News 3-13-2001, retrieved at http://news.bbc.co.uk/2/hi/business/1217716.stm

DPMA 1981. DPMA Model Curriculum, 1981. Park Ridge, Illinois: Data Processing Management Association.

DPMA 1986. DPMA Model Curriculum, 1986. Park Ridge, Illinois: Data Processing Management Association, 1986.

Gorgone, J.T., Davis, G.B. Valacich, J., Topi, H., Feinstein, D.L. and Longenecker. H.E. (2003). IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems. Data Base 34(1).

Gorgone, John T., J. Daniel Couger, Gordon B. Davis, David L. Feinstein, George Kasper, and Herbert E. Longenecker 1994. "Information Systems '95," DataBase, Volume 25, Number 4, November 1994, pp. 5-8.

Kautz, Karlheinz, Sabine Madsen, and Jacob Nørbjerg. "Persistent problems and practices in information systems development." Information Systems Journal 17.3 (2007): 217-239.

Landry, J. P., Longenecker, H.E., Haigood, B. and Feinstein, D.L.. 2000. "Comparing Entry-Level Skill Depths Across Information Systems Job Types: Perceptions of IS Faculty," Proceedings of Sixth Americas IT 2008. IEEE/ACM Joint Task Force on Computing Curricula.

Information Technology 2008, Curriculum Guidelines for Undergraduate Degree Programs in Information Technology, ACM and IEEE-Computer Society, November 2008. Retrieved at http://www.acm.org/education/education/curricula-recommendations

Longenecker, H.E., Feinstein, D.L., Couger, J.D., Davis, G.B. and Gorgone, J.T. (1995). "Information Systems '95: A Summary of the Collaborative IS Curriculum Specification of the Joint DPMA, ACM, AIS Task Force," Journal of Information Systems Education, Volume 6, Number 4, pp. 174-187.

Longenecker, H. E., Jr., D. L. Feinstein, J. D. Couger, G. B. Davis, and J. T. Gorgone (1995). Information Systems '95: A Summary of the Collaborative IS Curriculum Specification of the Joint DPMA, ACM, AIS Task Force. Journal of Information Systems Education, Volume 6, Number 4, pp. 174-187.

Longenecker, H.E., Feinstein, D.L. and Babb, J.S. (2013). Is there a need for a CIS Model Curriculum?, Proceedings of ISECON, San Antonia 2013.

Nunamaker, J.F., Couger, J.D. and Davis, G.B. (1982). "Information Systems Curriculum Recommendations for the 80s: Undergraduate and Graduate Programs," Communications of the ACM, Volume 25, Number 11, November 1982, pp. 781-805.

Segal, B (1995). "A Short History of Internet Protocols at CERN," CERN IT-PDP-TE, retrieved at http://ben.home.cern.ch/ben/TCPHIST.html

SBE Council, 2015. Small Business Facts and Data, retrieved at http://www.sbecouncil.org/about-us/facts-and-data/

Teichroew, D (1971). "Education related for the use of computers in organizations", CACM 14,9 (September 1971).

The Standish Group 1995. retrieved at http://www.projectsmart.co.uk/docs/chaos-report.pdf

The Standish Group Report 2001. "Extreme Chaos" retrieved at http://www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf

Topi, H., Valacich, J., Wright, R.T., Kaiser, K.M., Nunamaker, J.F., Sipior, J.C., and Vreede, G.J. (2010). IS 2010 Curriculum Guidelines for Undergraduate Degree Programs in Information Systems, Association for Computing Machinery (ACM), Association for Information Systems (AIS)", retrieved July 14, 2012: http://www.acm.org/education/curricula/IS%202010%20ACM%20final.pdf

WiseGEEK (2015)."What was the dot-com bubble?," retrieved at http://www.wisegeek.org/what-was-the-dot-com-bubble.htm

Figure 2

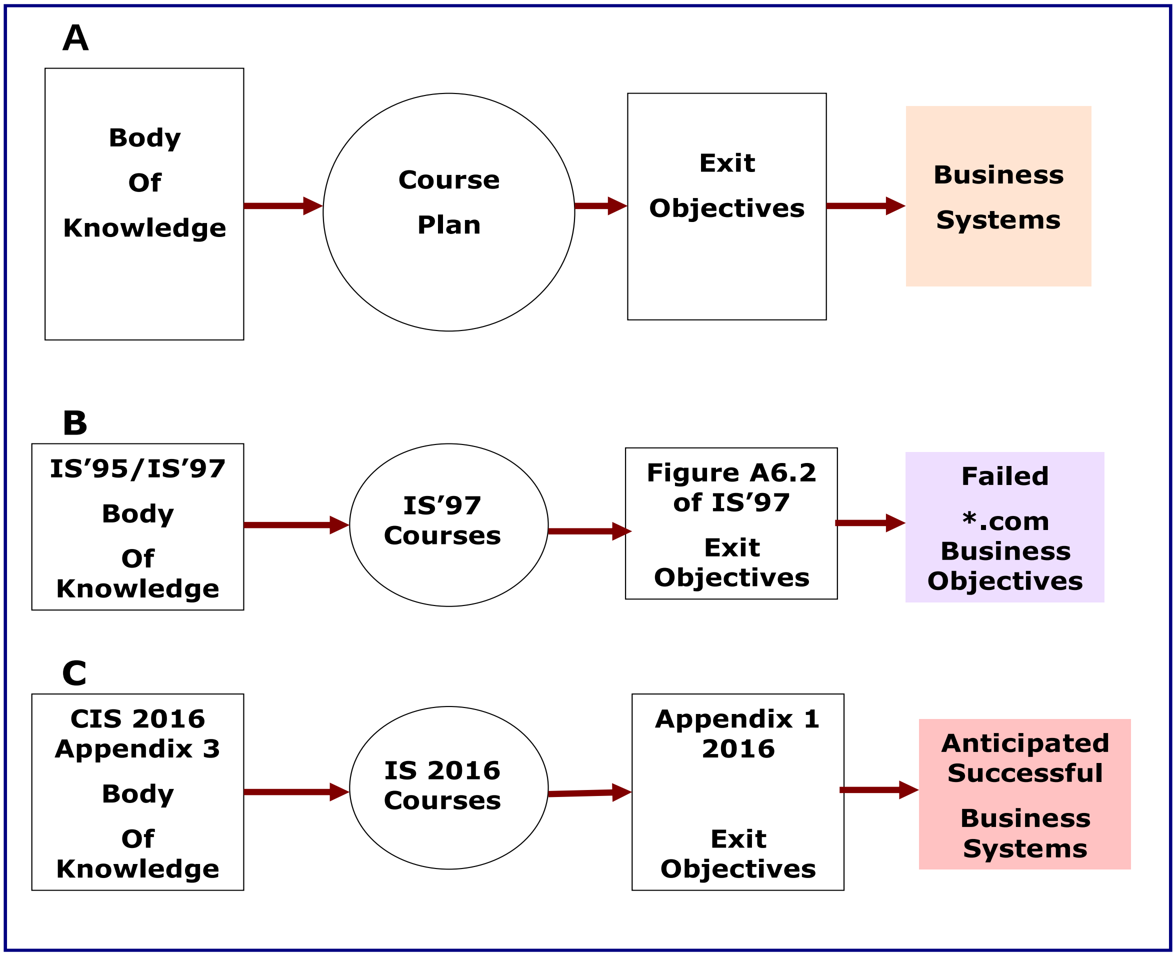


Figure 2. Curriculum Models: A) is a general model, B) is the model of IS'97 model, and C) is the Model of a to-be-proposed 2016 CIS model curriculum.

Appendix 1 – Successful Exit Objectives Statements Abstracted from The Standish Group (2001, 1995).

EO-1 Accurate business plan developed by end users, management and development team
- Identify stakeholders; ensure executive support
- Identify and qualify business knowledgeable project manager to deliver a competitive business plan
- Establish user—developer—management interactions to ensure involvement, and clear business objectives
- Choose a development methodology (e.g. Agile, Lean UX…)

EO-2 Exceptional requirements analysis
- Must use a User-Centered Focus
- Must express IT alignment with a high degree of maturity
- Identify Deployment System Requirements
- Must be tied to a verification and validation mechanism
- Must involve excellent team, personal and interpersonal skills

EO-3 Translation of Requirements info viable software
- Should consider using Agile approach
- Must use well established software engineering and programming practices, including reuse
- Must have exceptional database modeling and implementation skill
- Must apply quality principles

EO-4 Deploy Software Product
- Install system on IT host
- Test Software
- Test System and certify

EO-5 Project Management based on established formal written methodology
- Initiate project thoroughly understood by the project manager
- Establish project communication
- Must set important milestones and check points
- Perform project risk management
- Assure project/product security
- Utilize reusability
- Develop WBS tied to system development life-cycle
- Establish configuration management
- Execute project subject to quadruple constraint; minimize scope
- Use project control tools PERT/Gantt; requirement tools; collaborative tools

Appendix 2 – Knowledge Areas Sorted by Depth of Knowledge CIS Students Must Master in an Undergraduate Curriculum for Depth 2.5 or Greater

KA	Knowledge Areas Sorted by Expert Expected Depth of knowledge	Depth 0-5	Emphasis theory -50 practice +50	Expert Confidence
A22	Database	4.3	28	97%
B03	Data Retrieval and / Manipulation with Database Languages	4.0	26	93%
A23	Analysis and Specification / of System Requirements	4.0	19	90%
A19	Analysis of Business / Requirements	3.9	20	94%
A21	Information Systems Design /	3.9	21	93%
C01	Programming / Fundamentals	3.8	32	94%
A07	Web Systems and / Technologies	3.8	32	92%
A15	Approaches to Systems / Development	3.7	24	96%
G08	Project Plan, Scope, and / Initiation	3.7	21	90%
D14	Systems Analysis & / Design	3.6	23	92%
B05	Data and Information / Modeling at Conceptual and logical Levels	3.5	17	90%
C09	Software / Requirements	3.5	18	93%
A24	Team and Interpersonal / Skills	3.5	20	94%
G07	Leading Project / Teams	3.5	21	91%
B01	Database Systems and / Distributed Databases	3.4	18	86%
C15	Software / Design	3.4	20	90%
A06	Information Technology / Fundamentals	3.4	23	91%
D09	Systems Development / Concepts and Methodologies	3.3	15	91%
C05	Human Computer / Interaction	3.3	17	92%
C16	Software Development / Fundamentals	3.3	25	92%
B12	Data Integrity and / Quality	3.2	19	91%
A20	Information and Business / Analysis	3.2	15	90%
D11	Systems Implementation and / Testing Strategies	3.2	16	89%
C06	Module Design and / Construction	3.2	18	91%
C19	Software / Testing	3.2	21	91%
G10	Project Execution & / Control	3.2	20	87%
A01	Impact of Information / Systems on Organizational Structure and / Processes	3.1	15	91%
D06	System Deployment and / Implementation	3.1	17	89%
B07	Physical Database / Implementation / Data Definition Language	3.0	16	92%
A13	Business Intelligence and / Decision Support	3.0	14	90%
M03	Basic Scripting/ / Programming	3.0	25	92%
B04	Teams and Interpersonal / Skills	2.9	15	89%
M01	Basic Data / Analysis	2.9	21	86%
G12	Project / Quality	2.9	8	82%
C13	Security and Privacy, / Vulnerabilities, Risks, Mitigation	2.9	10	78%
B08	Stored Procedure / Implementation	2.8	20	85%
B10	Data and Database / Administration	2.8	18	88%
A03	Identification of / Opportunities for IT enabled Organizational / Change	2.8	10	92%
A16	Different Approaches to / Implementing Information Systems	2.8	15	91%
C02	Programming / Languages	2.8	14	92%
C17	Software / Construction	2.8	30	89%
G06	IS Project Strategy and / Management	2.8	8	85%
G03	Establishing Project / Communication	2.8	13	88%
G09	Work Break-down / Structure	2.8	18	90%
G13	Project / Closure	2.8	13	84%
E04	Networks and / Communications	2.8	15	81%
H02	Probability and / Statistics--Basic probability theory, random variables and / probability distributions, estimation theo...	2.8	8	82%

B11-	Data Management and / Transaction Processing	2.7	22	90%
A17-	Business Process Design and / Management	2.7	15	89%
A26-	Computer / Networks	2.7	21	89%
H01-	Math and Statistics for / IT	2.7	18	87%
D15-	User / Experience	2.6	16	89%
G11-	Project / Standards	2.6	11	83%
A02-	Individual and / Organizational Knowledge Work Capabilities	2.6	14	89%
C04-	Integrative Programming and / Technologies	2.6	18	78%
B13-	Security attacks and / mitigations	2.6	15	84%
B06-	Scripting	2.5	25	91%
A09-	Enterprise / Architecture	2.5	4	79%
D07-	System Verification and / Validation	2.5	12	86%
A25-	Configuration and Change / Management	2.5	12	85%
C03-	Programming / Environments	2.5	16	87%
C07-	Software Engineering / Process	2.5	15	89%
C18-	Software / Quality	2.5	11	88%
C20-	Software / Maintenance	2.5	17	89%
D05-	System Integration and / Architecture	2.5	13	83%
F02-	Information Assurance and / Security	2.4	3	78%

Knowledge Areas on Required Depth for an Undergraduate CIS Major (Longenecker H, et al, 2015.) The above is a partial list of knowledge areas 2.4 and above. These areas will involve the most effort through the cognitive load they demand. It is noteworthy that these knowledge areas map very closely to the demands of the Standish Group specification (see Appendix 1).

Appendix 3 – CIS Body of Knowledge (9-3-2015)

BK-1 Database
- Database Components (entities, attributes, relationships, drawing, scripting)
- Database Structuring (Create, Modeling, Quality, integrity, data types, data, and indexes)
- Database Access (DDL, DML, Transaction Processing, Stored Procedures; blocking injection attacks)
- Database Services (ETL, Report Services, BI, DSS, Backup, Replication, Security Management, Administration)

BK-2 Information System Development
- IS Development: Planning; intellectual purpose; Feasibility; privacy; security; alignment security
- IS Development: Make or Buy
- IS Problem Definition, Requirements Elicitation; BPR Analysis
- IS Organization Development with New IS (IT enabling, improved IT alignment, lower resistance, raise involvement)
- IS Design Maturity (levels within apprenticeship, design-leadership)
- IS System Verification/Validation Planning
- IS Development Test Plan
- IS Verification with Customer
- Assertion of Quality Policy
- IS Test and Validate (Module, Application, System)
- IS Final Evaluation, Deployment and Operation
- IS Team and Interpersonal Skills (Leadership, Empowerment, Change, Meetings, Teams, Innovative learning)
- IS Life Cycle Tools (Methodologies, Support Systems, Bloom and Learning for Clients and Students)

BK-3 Information Systems Design
- IS Design Architecture, Frameworks, Creativity, Reflection, video, voice
- IS Application Design (Requirements, Modules, Verification)
- IS Design Paradigms (cash management, new accounts, new addresses, new organization interaction, international actions, interfaces management, security procedure, Sarbanes Oxley, HCI management, HH device utilization)
- IS BPR, Data Transformations, Reporting, and BI
- IS Design Standards, Privacy and Security, Policies, Regulation and Compliance
- IS Design Quality (Verification and validation, qualitative and quantitative-assessments)
- IS Systems Testing and Implementation
- IS Configuration and Change Management

BK-4 Software Planning, Programming, Testing
- Programming Logic and Design (computers, programming, programs, control structures, sequence, selection, loop, arrays, records, modules, parameters, OO, events, files and DB)
- Programming Implementation (Languages, Environments, Compilers, Local, Web Environment; Code-a-little--test-a-lot; scripting)
- Languages (C++, C#, VB.net, Java Script, HTML, ASP)
- OO Programming (OO Structures, concepts, implementation with an IDE, testing
- Software Engineering (Requirements, Simple Algorithms and Data Structures, Modules, Box Structured Design, Programming, Quality)
- Software Implementation (Requirements, Design, Modular Top Down Implementation, Testing, Validation, packaging, installation, operation)
- Software Management (Development, Maintenance, documentation, standards, performance)

BK-5 IS Project Management
- Project Initiation (Strategy, Stakeholder analysis, Plan, Scope)
- Project Communication (Classification, Frequency, Responsibilities, Monitoring)
- Project Staff (Function, Responsibilities, Qualification, Reporting)
- Risk Management
- Work Break Down (Structure, Schedule--upcoming and completed events)
- Teaming (ensure team training, ensure leadership development, managing disputes)

Project Execution and Control (quadruple constraint, controlling activities, negotiating changes; ensure standards & quality; tools: Gantt, PERT)
Project Closedown (Acceptance Reviews, Final Reporting)

BK-6 Technology, IT Management and Security
IS Professionalism (systems thinking, organizational behaviors, legal issues, ethical issues, social issues, concepts of performance, practicing success habits, life-long learning)
Using IT governance (ITIL, regulatory standards, compliance)
IA Fundamentals (Vulnerabilities, Risks, Mitigation, threats, attacks, incident management, Security Policy Principles and Design)
Computer Architecture and Organization
Parallel and Distributed Computing
Devices (cable, fiber, modem, router, switch packet shaper, protocols, servers, sniffers)
Networks and Communications, security issues
Operating Systems Concepts, security issues
Storage management systems
System Operation, Administration and Maintenance
Virtualization and zero client

Appendix 4A – Body of Information Systems Knowledge for IS'95 and IS'97

1.0 Information Technology

1.1 Computer Architectures
1.2 Algorithms and Data Structures
1.3 Programming Languages
1.4 Operating Systems
1.5 Telecommunications
1.6 Database
1.7 Artificial Intelligence

2.0 Organizational and Management Concepts

2.1 General Organization Theory
2.2 Information Systems Management
2.3 Decision Theory
2.4 Organizational Behavior
2.7 Managing the Process of Change
2.8 Legal and Ethical Aspects of IS
2.9 Professionalism
2.10 Interpersonal Skills

3.0 Theory and Development of Systems

3.1 Systems and Information Concepts
3.2 Approaches to Systems Development
3.3 Systems Development Concepts and Methodologies
3.4 Systems Development Tools and Techniques
3.5 Application Planning
3.6 Risk Management
3.7 Project Management
3.8 Information and Business Analysis
3.9 Information Systems Design
3.10 Systems Implementation and Testing Strategies
3.11 Systems Operation and Maintenance
3.12 Systems Development for Specific Types of Information Systems

IS Body of Knowledge for IS'95 and IS'97. This body of knowledge was widely reviewed both by academic and industry professionals and became the basis for IS'95 and IS'97. This body of knowledge is at best partially supportive of Appendix 1. At best it represents a very incomplete match to the thought process that emerged and was specified by the Standish Group (1991, 1995).

Appendix 4B – Exit Objectives (from IS'97, Figure A6.2)

D. Systems Development

D.1 Software Development

Is Life Cycle: Developing With Packages
Implementing And Event Driven Applications
Is Application Development/Code Generate
Is Database And Is Implementation
Is Database Application Implementation
Is Database Application Structuring
Is Development Testing
Is Applications, Production Systems

D.2 Database

Database Terminology And Concepts
Implementing A Simple Database Design
Is Database Applications Development
Is Data Modeling
Is Database Conceptual/Logical Models

D.3 Sys Analysis/Design

Info Analysis: Individual Vs Group
Info Analysis: Finding Is/It Requirements
Is Development Standards
Is Development Risks/Feasibility
Is Conversion Planning
It Systems Specification
Problem Solving, With Packages
Is Analysis And Design Tasks
Is Continuous Improvement And Is
Is Design And Implementation
Is Rapid Prototyping
Is Requirements And Specifications
Systems And Quality Metrics/Assessment
Is Development And Conversion
Is Functional Specifications
Is Requirements And Database

D.4 Teams/Interpersonal

Interpersonal, Synergistic Solutions
Interpersonal, Consensus Development
Interpersonal, Group Dynamics
Interpersonal, Agreements/Commitment
Personal, Time/Relationship Management
Personal, Presentation
Interpersonal, Empathetic Listening
Interpersonal, Goal/Mission Alignment
Personal, Proactivity/Principled Action

D.5 Project Management

Is Development Project Close Down
Is Development And Project Management
Quality And Performance Management
Is Development Project Planning
Is Development Project Management
Is Development Project Management
Is Development Project Management Tools
Is Life Cycles And Projects

E. Is Deployment And Management

E.1 Support Services

Personal, Life-Long Learning

E.2 Systems Integration

Telecom, Installation, Implementation
Telecom, Lan, Install, Configure
Os, Install Multi-Media
Os, Interoperability And Sys Integration
Os, Install Multi-User System
Is Commercial Implementations

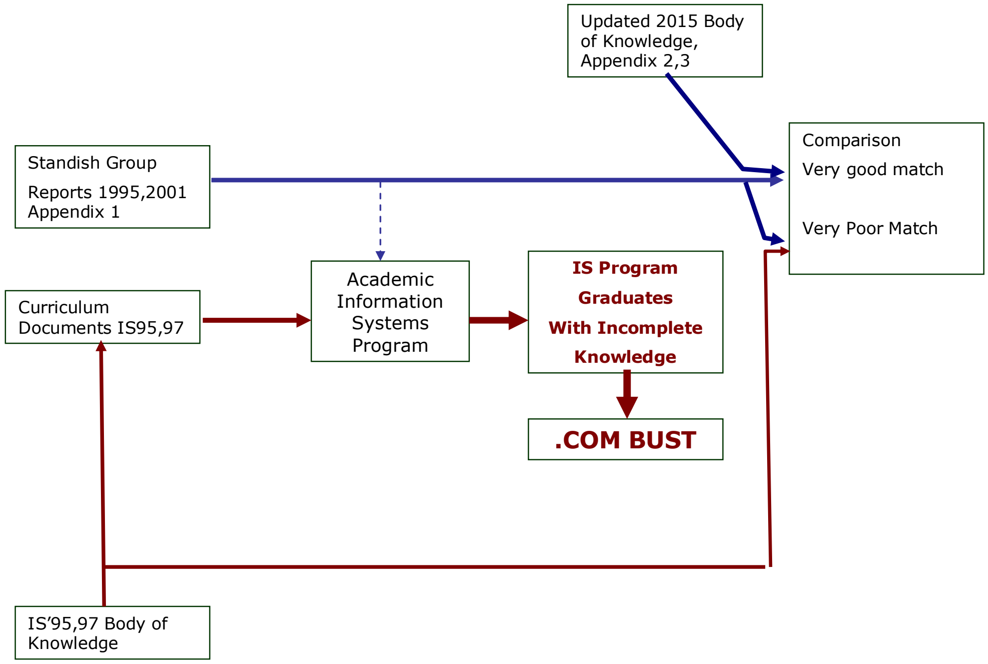
E.3 Management Of Is The Function

Is Careers
Personal, Performance Evaluation
Is Implementation, Outsourcing
Is Policies And Standards
Is Management And Dept Organization
Is Responsibility To Sell Designs To Mgt
Personal, Leadership And Is

E.4 Information Resource Management

Is Implementation And Outsourcing
Information Use Strategies

Appendix 5 – 2015 Updated BK Matches Standish Group Reports, yet a Poor Match is Obtained between IS95, IS97 BK.



Determine Curriculum Sufficiency by Mapping Body of Knowledge to Project Failure Statistics recorded and analyzed by the Standish Group Recommendations (1995, 2001). The .COM Bust occurred because of the incompetency of professionals. Ultimate success required a project manager skilled in business; the technical requirements of software design, programming, and database were necessary, but not alone sufficient!