INFORMATION SYSTEMS EDUCATION JOURNAL

Volume 16, No. 6
December 2018
ISSN: 1545-679X

In this issue:

iscap
Information Systems & Computing Academic Professionals

The **Information Systems Education Journal** (ISEDJ) is a double-blind peer-reviewed academic journal published by **ISCAP** (Information Systems and Computing Academic Professionals). Publishing frequency is six times per year. The first year of publication was 2003.

ISEDJ is published online (http://isedj.org). Our sister publication, the Proceedings of EDSIGCON (http://www.edsigcon.org) features all papers, panels, workshops, and presentations from the conference.

The journal acceptance review process involves a minimum of three double-blind peer reviews, where both the reviewer is not aware of the identities of the authors and the authors are not aware of the identities of the reviewers. The initial reviews happen before the EDSIGCON conference. At that point papers are divided into award papers (top 15%), other journal papers (top 30%), unsettled papers, and non-journal papers. The unsettled papers are subjected to a second round of blind peer review to establish whether they will be accepted to the journal or not. Those papers that are deemed of sufficient quality are accepted for publication in the ISEDJ journal. Currently the target acceptance rate for the journal is under 40%.

Information Systems Education Journal is pleased to be listed in the Cabell's Directory of Publishing Opportunities in Educational Technology and Library Science, in both the electronic and printed editions. Questions should be addressed to the editor at editor@isedj.org or the publisher at publisher@isedj.org. Special thanks to members of AITP-EDSIG who perform the editorial and review processes for ISEDJ.

INFORMATION SYSTEMS EDUCATION JOURNAL

Triangulating Coding Bootcamps in IS Education: Bootleg Education or Disruptive Innovation?

Leslie J Waguespack
lwaguespack@bentley.edu
Computer Information Systems
Bentley University
Waltham, Massachusetts 02452, USA

Jeffry S. Babb
jbabb@wtamu.edu
Computer Information and Decision Management
West Texas A&M University
Canyon, Texas 79016, USA

David J. Yates
dyates@bentley.edu
Computer Information Systems
Bentley University
Waltham, Massachusetts 02452, USA

Abstract

Coding bootcamps number in the hundreds world-wide despite repeated predictions of their demise over the past few years. Fueled by a resurgent economy and a persistent shortage of app developers and computer systems engineers, bootcamps tout a fast-track to a six-figure salary for as little as one-eighth the tuition dollars or time investment of a nominal four-year information systems baccalaureate degree. Bootcamps represent an enticing opportunity for: a) high school graduates unconvinced of the return on the time and money investment in a liberal arts education, b) college graduates who find their career potential limited by their baccalaureate major, or c) experienced workers seeking a change of profession. Although potentially disruptive, and generally neither accredited nor affiliated academically, bootcamps introduce opportunities for innovation in terms of structure, organization, curriculum, and pedagogy for traditional computing education in higher education, which we explore in this paper.

Keywords: IS education, Coding bootcamps, IS curriculum, IS workforce preparation

1. INTRODUCTION

The emergence of coding bootcamps is due in part to the shortage of computing professionals graduating from universities and the broad demand for individuals with hands-on software skills (Geron, 2013). According to Wikipedia, these bootcamps "provide a vocational training for free or a fraction of the cost of a college degree and are a part of the 'Edtech Disruption of Higher Education'" (Wikipedia, 2017b, p. 2). In addition to being less expensive than a college degree, coding bootcamps take less time by delivering an immersive learning experience, often in 8 to 12 weeks, after which students have learned how to code in a specific domain, e.g., web or mobile software development. Some programs even go into more depth within a domain, e.g. front-end development, iOS, Android or cloud-native development, and some offer a portfolio of such programs. Since most students prefer to learn as part of a community, especially during an immersive (and intense) experience, there are many more classroom-based than on-line bootcamps. Employing in-person cohorts like their military namesake, they offer emotional and psychological support that engenders a sense of confidence and professionalism (Barnett, Basom, Yerkes, & Norris, 2000). And, presumably for job placement reasons, these programs tend to cluster in population centers with a significant presence of technology companies. In the United States, for example, many of the well-known bootcamps have classrooms in San Francisco and New York. Recent diversification away from "just coding" bootcamps has given rise to camps focused on applications, e.g. data analytics, and infrastructure, e.g. Internet of Things.

So, since their inception in 2012, how are these alternative education programs doing? A survey of most U.S. graduates conducted by SwitchUp.Org (2017a), draws the following conclusions based on data gathered from 2014 to 2016:

- 63% of code bootcamp graduates reported increase in salary (in 2016 the average annual salary increase six months after graduation was more than $22,000);
- 80%+ of graduates were satisfied with their bootcamp education (just under 15% were dissatisfied);
- Average class size is 30 with a 1-to-3.8 student instructor ratio;
- Coding bootcamps are a far cheaper and accelerated option than learning to code at a university (the average bootcamp took 10.8 weeks in 2016 and cost $12,800);
- Women learning how to code represent 43% of the bootcamp alumni; and
- The bootcamp market is growing rapidly, projected to double from 2016 to 2017.

This report goes on:

> "There is no doubt that 21st century technology education is trending towards transparent, outcome-driven metrics. … However, key questions remain: Can the type of salary increase seen from the data be sustained in the long-term? As the supply of developers increases to match the demand, will the job market get tighter, or will the creation of tech jobs continue to outmatch the supply of developers over the next few years?" (SwitchUp.Org, 2017a, p. 7)

2. WHAT IS A BOOTCAMP?

Coding bootcamps offer technology-focused training programs that teach programming, frameworks, systems and tools which are in demand in many entry-level software developer positions. Most of these programs teach people with little or no technical coding background how to code, build and deploy applications.

Most information systems and computer science students spend four years to complete their degree. Code bootcamps are designed to distill skills from a four-year degree that are in the greatest market demand and infuse them with relevant methodologies and practices to bridge the perceived gap between contemporary academia and the real world of professional coding (Janicki, Cummings, & Kline, 2014; Yourdon, 2002). With an average program duration of less than 11 weeks, this requires a combination of a singular focus on high demand skills and technologies and high-impact learning with no frills.

As for colleges and universities, there are differences in how different coding bootcamps teach and prepare their students to enter the technology workforce. Because bootcamps lack oversight by federal and state governments or by accrediting bodies, any assessments or judgements about their quality are largely anecdotal. Many differentiating themes that emerge in both favorable and unfavorable anecdotes, however, are familiar to the EDSIG membership and EDSIGCON audience:

- Quality and focus of the curriculum;
- Technical training and know-how of instructors;
- Number of full-time vs. part-time instructors;
- Quality of instruction;
- Emphasis on group projects that simulate real-world development; and
- Availability of mentorship and tutoring for students.

3. BOOTCAMPS AS COMPUTING PROGRAMS

If nothing else, coding bootcamps represent a distinct departure from the prevalent models of career preparation followed by tradition institutions of higher education. A technology focus is obvious in a 2017 ranking of coding bootcamps by an industry monitoring website, SwitchUp.org (2017b), that identifies "The Best of 2017." Table 1 lists their ranking of 31 coding bootcamps and the "catalog" of technology training advertised by each.

Ranking	SWITCHUP™ 31 Top Rated Boot Camps	JavaScript	HTML	CSS	Ruby / Rails	ExpressJS/Node	AngularJS	Front-End Web	Full-Stack	UX Design	JQuery	Mobile App	SQL	Git	Swift	iOS	MongoDB	React.js
		Technology Training Most Frequently Offered																
1	App Academy	■	■	■	■								■					■
2	Le Wagon	■	■	■	■													
3	Bloc	■	■	■	■		■			■					■	■		
4	Ironhack	■	■	■	■	■	■										■	
5	Startup Institute																	
6	Flatiron School	■	■	■					■						■	■		■
7	Epicodus								■	■		■						
8	Actualize	■	■	■	■								■	■				
9	Founders and Coders	■				■												■
10	Dev League	■	■	■		■	■				■							
11	Designation									■								
12	Fullstack Academy	■	■	■		■					■							■
13	V School							■	■	■		■						
14	The Software Guild								■			■						
15	Codeup								■									
16	Grand Circus							■	■			■						
17	Starter League	■								■								
18	Tech Talent South	■	■	■	■													
19	Makers Academy													■				
20	Code Fellows	■						■	■			■				■		
21	Code Foundry	■	■	■			■					■						
22	The Iron Yard	■	■	■		■		■			■		■	■				
23	Dev Mountain	■				■	■			■	■				■	■	■	
24	Thinkful	■	■	■		■	■			■					■			
25	Lighthouse Labs	■	■	■	■	■	■								■			
26	Launch Academy	■	■	■	■			■					■	■				
27	Hack Reactor	■	■			■					■		■	■			■	
28	Coding Dojo	■	■	■	■	■	■	■			■						■	
29	Galvanize	■	■	■				■										
30	Dev Bootcamp	■	■	■	■								■					
31	General Assembly	■	■	■	■			■	■	■								
	# of These Camps Offering	23	19	18	11	10	8	8	8	8	6	6	6	5	5	4	4	4

Table 1. Code Bootcamp Technology Courses

In contrast, even the most technically oriented academic programs in colleges and universities require a significant investment of time and effort to develop a "liberally educated" citizenry. The Association of American Colleges and Universities expresses this model of liberal education thusly:

> "An approach to college learning that empowers individuals and prepares them to deal with complexity, diversity, and change. This approach emphasizes broad knowledge of the wider world (e.g., science, culture, and society) as well as in-depth achievement in a specific field of interest. It helps students develop a sense of social responsibility; strong intellectual and practical skills that span all major fields of study, such as communication, analytical, and problem-solving skills; and the demonstrated ability to apply knowledge and skills in real-world settings." (AACU, 2014, p. 1)

The AACU reported that 74% of surveyed employers in 2013 recommended this model of liberal education for college-bound students. Achieving the breadth of study ascribed to a liberal education involves on the order of 120 to 140 academic credit hours. Each credit hour unit translates into 15 hours of class time and 30 hours of student preparation, according to the U.S. Department of Education, International Affairs Office (USDoE, 2008).

In Table 2 that follows, the contrast between curriculum models, a typical twelve-week coding bootcamp versus the IS degree programs targeting two-year associate and four-year baccalaureate degrees, is dramatic (NCES, 2017).

Also, the difference in the student's overall program cost is significant, 1/4th to 1/8th the cost of an associate program or 1/5th to 1/10th the cost of a baccalaureate program. But, the most compelling differences are entailed by the singular focus of bootcamps on software development, programming. By eschewing the breadth aspect of the *liberal education*, the bootcamps typically require no study of the humanities, sciences, or post-secondary mathematics. In contrast to college programs in IS, by concentrating exclusively on code development and technical IT skills, the bootcamp applies virtually all the contact hours of instruction to coding related topics: more than twice the contact hours typically devoted to coding in the associate degree and nearly 80% more than in the baccalaureate degree. Furthermore, the bootcamp requires only about three calendar months rather than four or eight twelve-week semesters spanning two to four years for the associate or baccalaureate programs, respectively.

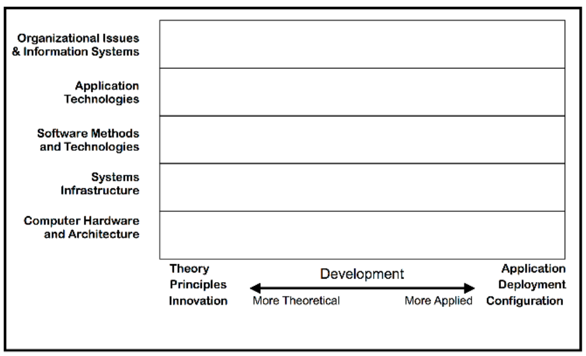
(Comparative Dimension)		Coding Bootcamp		Associate IS Program		Baccalaureate IS Program
Program Duration (typical)		3 month		2 year		4 year
Terms per program		1		4		8
Courses per term		6		5		5
Courses per programs		6		20		40
Credit hours per course		3		3		3
Credit hours per Program		18		60		120
Contact Hours per Credit		80		45		45
Curricular Coverage (typical)	# credits		# credits		# credits	
Coding Instruction	17	1360	9	405	12	540
Data Management	0.5	40	6	270	6	270
Development Theory	0.25	20	1	45	6	270
Computing Infrastructure	0	0	3	135	6	270
Application Domain Instruction	0.25	20	12	540	30	1350
Team Theory	0	0	2	90	9	405
Quantitative Methods	0	0	9	405	12	540
Humanities	0	0	9	405	24	1080
Sciences	0	0	9	405	15	675
Total Contact Hours of Instruction		1440		2700		5400
Low End Program Cost		$10,000		$40,000		$80,000
Program Cost per Hour		$7		$15		$15
Tuition		$7,000		$16,000		$32,000
Estimated Room & Board, Etc.		$3,000		$24,000		$48,000
High End Program Cost		$25,000		$120,000		$240,000
Program Cost per Hour		$17		$44		$44
Tuition		$17,000		$90,000		$180,000
Estimated Room & Board, Etc.		$8,000		$30,000		$60,000

Table 2. Bootcamp vs. IS Program Comparison

Triangulating Coding Bootcamps in the Curricular Geography of CC2005

The singularity of focus that bootcamps exhibit is further demonstrated in the curricular focus within the domain of computing education. The Association for Computing Machinery, ACM, and the Institute for Electrical and Electronics

Engineering, IEEE, have consistently worked to normalize the structure and evolution of computing education through a series of published curricular guidelines for particular computing disciplines (CS, CE, IT, IS, and SE), as well as mapping the overall landscape as it did with Computing Curriculum 2005, CC2005 (Shackelford, McGettrick, Sloan, Topi, Davies, Kamali, Cross, Impagliazzo, LeBlanc, & Lunt, 2006, pp. 6-21). In that CC2005 report (the most recent and comprehensive cross-discipline analysis), the task force created graphic characterizations of "what students in each of the disciplines typically do after graduation." Each discipline is portrayed on a field of competency as a "footprint" of proficiency gained by completing the respective academic program. (See Figure 1.)



Graphic characterization of what students in this discipline typically engage.

Figure 1 - CC2005 Field of Computing Competency

The field of competency delineates computing activities ranging on the Y-axis from hardware issues on the bottom to organizational policy and information management at the top. The X-axis depicts purely *applied* involvement in computing activities to the far right to purely *theoretical* engagement of computing topics to the left.

To emphasize the degree of abstract conceptualization required to bridge between the physical and social world of computing as depicted along the vertical dimension, we superimpose the semiotic ladder as exposed by the footprint of the respective CC2005 discipline. A semiotic framework explicates the expression and transmission of ideas, knowledge, and meaning through human communications (Liu, 2000; Stamper, 1973, 1988). (See Figure 2.) Ascent along the Y-axis of the field of competency (Figure 1) entails a progressive amplification of domain modeling skills and contextualized interpretation requiring a commensurate proficiency in dealing with the complexity of the social context.

Semiotic Ladder	Semiotic Layer Description
Social World	Beliefs, expectations, functions, commitments, contracts, law, culture
Pragmatics	Intensions, communications, conversations, negotiations
Semantics	Meanings, propositions, validity, truth, signification, denotations
Syntactics	Formal structure, language, logic, data, records, deduction, software, files
Empirics	Pattern, variety, noise, entropy, channel capacity, redundancy, efficiency, codes
Physical	Signals, traces, physical distinctions, hardware, component density, speed, economics

Figure 2 - Semiotic Continuum of Constructs

The framework (aka semiotic ladder) depicted in Figure 2 orients and categorizes contextual concerns spanning the sociological and technological landscape that information systems design practice must navigate. The "ladder" represents layered abstractions progressing continuously from bottom to top, anticipating components both material and conceptual arranged as layers of scaffolding one atop the other. Each layer anticipates building blocks in a gradient of abstraction, a vocabulary of metaphorical constructs. Each layer is reminiscent of a virtual machine encapsulating the details of the supporting layers to present a homologous array of structural and behavioral resources upon which to examine the dialog between IS developers and IS consumers.

Although our discussion focuses on the jusxtaposition of coding bootcamps and IS education, we include the footprint of computer science in Figure 3 as an orienting reference point. CS graduates may be engaged in purely theoretical work ranging from efficient utilization of hardware components to systems management supported by machine learning. CS graduates are not generally engaged in off-the-shelf systems deployment or configuration. They are seldom responsible for organizational policy or design of low-level hardware for information infrastructure.

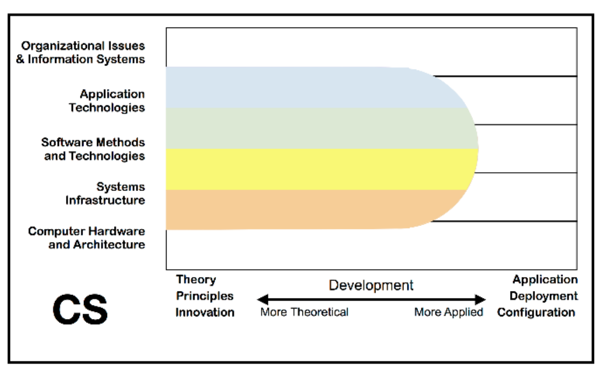


Figure 3 - Competency Target of CS (2005)

In CC2005's characterization of the activity of IS graduates, the full breadth of organizational information management policy and operational systems management appears without a significant involvement in hardware or software development theory and practice. (See Figure 4.) Software development is confined to applications and the configuration and deployment of off-the-shelf computing resources focussing largely on supporting business policies and functions.

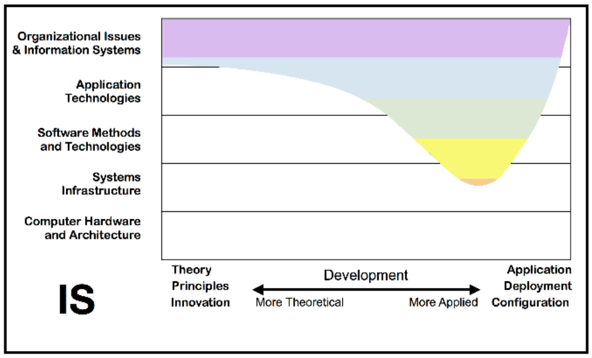


Figure 4 - Competency Target of IS (2005)

In our interpretation of the most recent IS curriculum guideline, IS2010, that task force appears to have interpreted the engagement of IS graduates as receding from direct engagement in software development by assuming a more consumer relationship with software systems (Topi, Valacich, Wright, Kaiser, Nunamaker, Sipior, & Vreede, 2010). (See Figure 5.) The task force appears to have envisioned IS graduates more focused on business systems as operational support by adopting a greater dependence upon third-party or out-sourced systems development rather than as builders themselves of strategic artefacts of business.

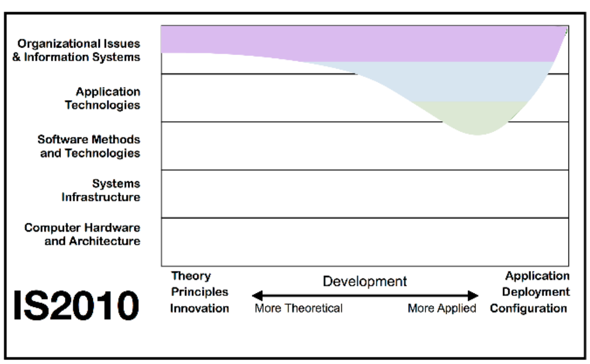


Figure 5 - Competency Target of IS2010

Bootcamps have never been a curricular focus of the ACM or IEEE guidelines efforts in the mode of CC2005. However, we posit here a footprint depicting the competency target of coding bootcamps in Figure 6 as a means to visualize aspects of the relationship between bootcamps and IS curricula.

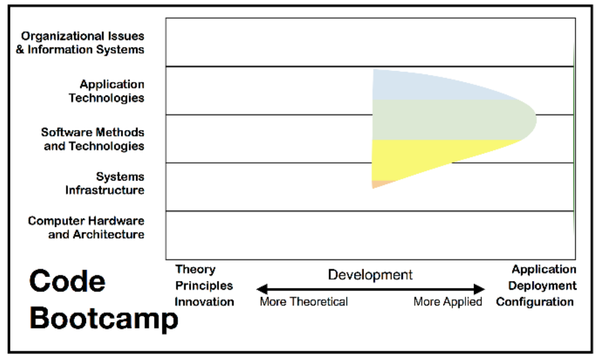


Figure 6 - Competency Target of Bootcamps

Bootcamps are purposefully quite "single-minded." They focus on individual computing technologies confined primarily to software production. Their goal is skill-building, rather than problem shaping or theorizing. The problem environment is usually fixed in terms of technology, platform, and tool set. The bootcamp goal is to produce an efficient, reliable, "construction" worker. While there is likely significantly more room to extend professional skill in the realm of software construction, it is clear that industry needs people who can "hammer nails" and "saw wood" rather immediately, hence the bootcamp phenomenon.

Bootcamps vs Accredited Curricula and Programs

While the value of accreditation in higher education is the subject of disparate opinions, nonetheless institutions, schools, programs and curricula each can be (and are often) accredited and such accreditation becomes a mark of quality for various parties: governments, industry, consumers, and citizens (Eaton, 2000, 2012).

As this paper is targeted to faculty in information systems (IS) and computing disciplines, we specifically reflect upon the influence and impact of both AACSB and ABET accreditation. While it is not the case that all IS programs would be either housed in a college of business, nor would they necessarily be accredited by either AASCB (for business) or ABET (computing), these accreditation bodies serve as reasonable proxies by which we may understand the influence that these, and regional and national accreditations, have on curricula and the programs that deliver them.

In the case of AACSB, accredited schools are asked to articulate, measure, evaluate, and improve upon key learning goals for all of programs that fall under the auspices of accreditation such that specified standards are met and maintained. The standards, as is often the case also with regional accreditations, apply to the breadth of activities that extend beyond curriculum. However, the direct and indirect impacts of these standards on curriculum are certainly an intentional byproduct (Gray, Smart, & Bennett, 2017; Solomon, Scherer, Oliveti, Mochel, & Bryant, 2017). Nonetheless, the guidance for curricula, as a component of learning and teaching, are general and broad. Thus, AACSB will examine the processes that lead to a curriculum that focuses on relevant skills and knowledge expected of a particular degree program, any specifics are left to faculty execution of their processes. Thus, while the 2017 specification of AACSB Standard Eight requires an articulation of learning goals which are mapped into course content whereupon some assurances of learning are adhered in a process of curriculum management, these are processes without specificity of content. AACSB Standard Nine provides some expectation that content is consistent with what is normative to a degree program – citing a requisite to care for theories, ideas, concepts, skills, and knowledge, these are to be established in the college.

As many Information Systems curricula have some organizational component, the general business knowledge areas specified by AACSB in Standard Nine would naturally cover some portion of what can be articulated as an Information Systems curriculum. The ACM and AIS Curriculum Guidelines for Undergraduate Degrees in Information Systems (Topi et al., 2010) is the latest installment in a long line of IS curricula guidelines designed to fill in the gaps which, particularly those topical to computing, are not addressed by AACSB. Such model curricula espouse principles regarding what a "standard" IS curriculum might look like while also leaving space for local specializations and adaptations. While not without some controversy regarding the degree of specification of technology content (Longenecker, Feinstein, & Babb, 2013; Reynolds, Adams, Ferguson, & Leidig, 2017; Waguespack, 2011), IS2010 made some clear vital elements, such as data and information management, infrastructure, and Systems Analysis and Design, among others.

With respect to IS curricula, ABET also provides guidelines for programs seeking to acquire and maintain a program-level accreditation. The specifics of the ABET's Computing Accreditation Commission (CAC) extend beyond that provided in the IS2010 report, while perhaps providing less justification and philosophy behind the specifics. The 2017-2018 CAC criteria specify both content – one year or, typically, 10 courses that cover basic content such as: coverage of the fundamentals of application development, data management, networking and data communications, security of information systems, systems analysis and design and the role of information systems in organizations. Within that year's coverage is included advanced coursework to extend these fundamental topics, coverage of a professional environment in which information systems will be applied – often in business - and also quantitative methods and statistics.

The cross-verifying (and validating) and interleaving nature of these externally-validated accreditations on IS curricula are clear. What is less clear is the degree to which bootcamps are providing similar, if not better, grounding in the technical components of an IS curriculum. While the advantages of a college education, even in computing, are somewhat established in the marketplace (Carnevale, Cheah, & Strohl, 2013), it is reasonable to ask what advantages coding and technology bootcamps pose? This question is particularly poignant as there is growing evidence that the labor market may not continue to give preference to the fruits of "traditional" higher education over two-year degrees, diplomas, certificates, MOOCs, and now coding bootcamps (Jepsen, Troske, & Coomes, 2014).

Deciphering a Bootcamp Advantage

It is fair to say that bootcamps are dedicated to providing the maximum of "knowing how" with the minimum of "knowing that" with virtually no attention to "knowing why" (Claxton, 1997)! To achieve their teaching goal of "knowing how," bootcamps employ three tactics: a) topic isolation, b) cohort cohesion, and c) practice immersion.

Topic Isolation: Unlike college or university philosophies that blend a disciplinary focus into a context of liberal studies, bootcamps identify and isolate their curriculum and pedagogy concentrating on the tools and skill set of a niche software development task domain. Common domains are website development, client side or server side programming, mobile device apps, and platform-based application development environments (e.g., LAMP Stack, Ruby on Rails, JavaScript, Java, C#, HTML, CSS, ASP.NET, Python, Swift, iPhone, Android, etc.). (See Table 1.)

Cohort Cohesion: The bootcamp environment engages the group-learn ethic of its military name-sake. Working shoulder-to-shoulder with classmates who virtually all are aiming at the identical academic and tactical goal of IT employment, students gain comrades and competitors with whom and from whom to learn, and draw energy to hold fast to the intense and often grueling 40-hour-plus class weeks. The group familiarity gained in the early weeks of the bootcamp foreshadow the proximity that *industrial-strength* development experiences will engender. At the same time, cohorts offer opportunities to learn team communication and leadership lessons unscripted in the bootcamp curriculum.

Practice Immersion: The typical 40-hour class week provides the close-up demonstration of *introduction to explanation to demonstration to exercise to evaluation* in a cycle that within a cohort provides an academic variant of *close order drill*, "the memorizing of certain actions through repetition until the action is instinctive to the soldiers being drilled." (Wikipedia, 2017a) At the same time there is the opportunity in the presence of the instructor to immediately validate understanding of the introduction and explanation by seeing the technology demonstrated as implemented and then engage a development behavior to replicate the implementation. All this pedagogy proceeds while suppressing the disruptive intervention of days separating class sessions or attention distracted by the study of topics other than the technology subject at hand. These characteristics may accrue advantages that are worthy of further examination in our own community.

4. EXPLOITING BOOTCAMPS AS I.S. CURRICULAR RESOURCES

The natural reaction of college computing programs to coding bootcamps might be to "man the bulwarks" and mount maximum resistance to their rising popularity. Or, higher education might "write off" bootcamps as a philosophically inferior approach to education. But honestly, bootcamps pose a tempting alternative for career entry to the computing profession – not only to the student market, but also to a parental and legislative audience growing skeptical of the cost / benefit or return on investment of traditional higher education.

The fact is that for some time now, there persists a demonstrated shortfall of skilled software developers in the job market (Geron, 2013). Most academics would consider bootcamps a myopic choice, but, bootcamps can equip a committed high school graduate, disillusioned liberal arts degree holder, or a working professional tired of their current career the opportunity to enter the computing career field. But, is there an opportunity for academic programs, particularly IS, to take advantage of this emerging model of programming education?

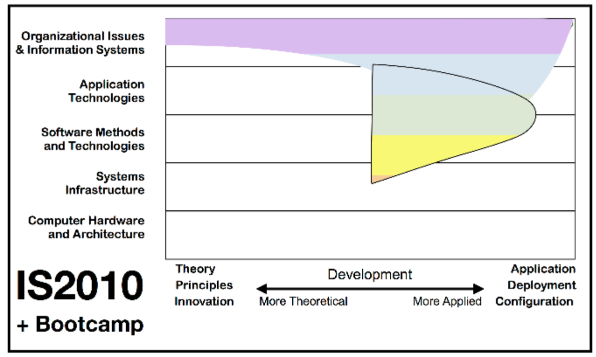


Figure 7 - Competency Target of IS (2010) + Code Bootcamp

A quick overlay of the posit we offer for the bootcamp footprint of competency onto that of IS2010 indicates that there is relatively little redundancy. (See Figure 7.) In fact, the combination is reminiscent of the IS footprint of CC2005, suggesting that perhaps some opportunities for curricula innovation present themselves.

IS Graduates Need Development Skills

Most IS programs envision their graduates' career entry into computing aligned with, if not embedded in, software systems development. To that end, even with the departure of software development requirements from IS with the IS2010 guidelines, most undergraduate IS programs today find it imperative to offer at least enough software development coursework to legitimize a place for that skill on their graduate's résumé. Relatively few IS graduates will place in positions that are primarily managerial or supervisory without some experience with programming responsibilities. Therefore, training for software development skills remains for the foreseeable future as requisite to career entry for IS graduates.

Teaching Coding Skills Costs IS Twice

Supporting software development coursework is doubly expensive for IS programs:

a) Consuming precious credit hours squeezed into business school programs dealing with the pressures for maintaining breadth in liberal arts within the strictures of business program accreditation; and,

b) the complexities of software development instruction that levies on faculty a burden of technical preparation and individualized student engagement that are not easily aligned with the models and areas of research promoted as flagship academic scholarship.

Search for a Win-Win Situation
Exploring ways to coopt bootcamps that teach coding skills may be mutually beneficial if they can: a) provide superior coding skill outcomes for students compared to the limited curricular resource for it in college and university programs, and b) lifting the training burden from IS faculty struggling to maintain a successful balance of teaching and research, both of which are grounded not in the computing but rather, the business disciplines where the primary standards of faculty evaluation reside. Some possible approaches are outsourcing software development skills training by accepting bootcamp completion for college credit as liberal arts coursework or as fulfilling some other distribution requirement, or insourcing the training as a summer intensive offering by the college. The latter might use underutilized housing and laboratory facilities and be staffed by a combination of practicing professionals, accomplished upper class students, and supervisory staff.

Teaming Up to Address the Skills Gap
Articulation agreements between bootcamps and IS programs can function as bilateral recruiting functions. Bootcamps can recommend IS programs for degree completion once they reach transition points in their development careers. And, colleges can recommend bootcamps as “test drives” for undecided students unsure of the two-year or four-year commitment to college. In either case, local businesses strapped by a shortage of programmers and app developers may want to explore internship, scholarship and mentorship arrangements to access the best and brightest prospects. These businesses may want to influence the bootcamp curricula regarding tools and skills appropriate for their information technology strategies, as well as, opportunities to upgrade or retrain the skill sets of their current employees.

5. DISCUSSION

Our exploration of coding bootcamps is not intended to malign or endorse the phenomenon, but rather to consider the challenges and opportunities. To summarize, we conclude with a simple SWOT analysis.

Strengths. We have elucidated the strengths of the code bootcamps as being very focused on specific technologies which are immediately valued and favored in the marketplace. Often located in population and technology hubs, the camp-to-employer food chain is compelling for the employer. These are fresh students who are ready to go with the timely skills required at an entry level. With career-switchers, employers get some of the polish and seasoning of work experience, which is generally favored in most industries evidenced in a lower unemployment rate for those with experience in almost any industry (Jepsen et al., 2014).

Weaknesses. Relative to the long-standing inertia of experience that traditional college-oriented programs and curricula in computing enjoy, there will likely be a wide range of providers and standards (or lack thereof) as the code bootcamp innovation diffuses and competition among providers increases. With no oversight, these bootcamps already deliver up mixed results with little recourse for students that feel short-changed. US Department of Education actions sanctioning ITT Technical University for fraudulent practices may be a cautionary tale here as we have witnessed some drawbacks in for-profit higher education (Morey, 2004).

Opportunities. As we have indicated earlier, two-year, four-year, and graduate institutions have the longstanding expertise in providing effective instructional environments. While many of these coding bootcamps are fitting in where they can, including dedicated commercial office spaces, institutions of higher education remain nexus points where a crossroads of research, instruction, technology, employers, and students can comingle. Rather than remaining averse to technology-wrought emerging models for instruction and learning (Hanna 1998; Hamilton 2016), institutions of higher education may do well to integrate this mode of delivery to realize its advantages and capitalize on the industry connections inherent in the code bootcamps. Often, higher education institutions are responsible for relationship building, an experientially-rich learning environment, and the maturation of students – particularly those of traditional college age. Regardless of what professional stage the code camper is, higher education can embrace elements and aspects of how programming, and other IT work, is increasingly seen as the "next blue-collar job" (Thompson, 2017). This is against a backdrop of futurist, near-desperate vision regarding a lack of employment opportunities in the face of automation, machine-learning, artificial-intelligence, and robotics (Clark, Graham, & Jones, 2017).

Threats. Perhaps of most interest to the IS academic would be appropriate questions about how/whether coding bootcamps will disrupt the market share that IS programs hold. In the "dot bomb" era, many sought out certifications and degrees from two-year and four-year institutions and any other means to get on the bandwagon of a super-heated bubble (Yourdon, 2002). Much has changed since that era. One change is the cocktail of outsourcing, offshoring, near-shoring, and on-shoring that pervades the labor market in software and systems development (Worley, 2012). Another is the advent of MOOCs and open education (Yuan & Powell, 2013). Further, the continued advances of service-dominant logics, Web 2.0, the Internet of Things, Social Media and Video Sharing create a new mix of information and learning vectors. Consider an event held in May of 2017 in Prague, Czech Republic, a country known for providing talent in business process and technology outsourcing, called Jobs Dev 2017 (Layman, Williams, Damian, & Bures, 2006; https://www.jobsdev.cz/). As an intersection to "facilitate developer-to-developer dialogue and offer a place where companies from a wide range of IT industries can meet with skilled programmers, freelancers, developers, and university graduates," this may represent an emerging trend where entry-level, mid-level, and senior-level talent can meet directly with employers. While any decent university job-fair would create the same facilitative environment, what if these meetings create a reality where the university is the unnecessary "middleman?" Increasingly, Codecademy, CodeHS, Coursera, Khan Academy, Lynda.com, and Udacity, among others, each can provide effective and focused instruction in the entry-level skills that get jobs, jobs that graduates of information systems programs are also vying for.

What if these new outlets will do a better job of teaching hands-on skills? How might we join, coopt or lead in this new environment? In fact, are we even now being left behind? Some information systems education researchers already seem to think so (Burns, Gao, Sherman, Vengerov, & Klein, 2014; Janicki et al. 2014). This paper invites continued inquiry and discussion regarding the coding bootcamp phenomenon.

6. REFERENCES

AACU. (2014). Association of American Colleges and Universities, retrieved July 7, 2017, www.aacu.org/leap/what-is-a-liberal-education#survey

Barnett, B. G., Basom, M. R., Yerkes, D. M., & Norris, C. J. (2000). Cohorts in educational leadership programs: Benefits, difficulties, and the potential for developing school leaders. *Educational Administration Quarterly*, 36(2), 255-282.

Burns, T. J., Gao, Y., Sherman, C., Vengerov, A., & Klein, S. (2014). Investigating a 21st century paradox: As the demand for technology jobs increases why are fewer students majoring in information systems? *Information Systems Education Journal*, 12(4), 4.

Carnevale, A. P., Cheah, B., & Strohl, J. (2013). Hard times: College majors, unemployment and earnings: Not all college degrees are created equal, retrieved July 7, 2017, cew.georgetown.edu/wp-content/uploads/HardTimes2015-Report.pdf

Clark, J. W., Graham, C. M., & Jones, N. (2017 August). Information systems and the problem of work: Protocol for a systematic review. Proceedings of Americas Conference on Information Systems (AMCIS 2017), Boston, MA.

Claxton, G. (1997). Knowing without knowing why, *The Psychologist*, May 1998, 217-220.

Eaton, J. S. (2000). Core Academic Values, Quality, and Regional Accreditation: The Challenge of Distance Learning. Council for Higher Education Accreditation, Washington, DC.

Eaton, J. S. (2012). The future of accreditation. *Planning for Higher Education*, 40(3), 8.

Geron, T., (2013, February 26) "Bill Gates, Mark Zuckerberg, Chris Bosh campaign for more programmers," Forbes, retrieved July 7, 2017, www.forbes.com/sites/tomiogeron/2013/02/26/bill-gates-celebrities-support-education-for-computer-programming/#5b21d60f7ff8

Gray, D. M., Smart, K. L., & Bennett, M. M. (2017). Examining espoused and enacted values in AACSB assurance of learning. *Journal of Education for Business*, 1-7.

Hamilton, E. (2016). Technology and the Politics of University Reform: The Social Shaping of Online Education. Palgrave Macmillan, New York, NY.

Hanna, D. E. (1998). Higher education in an era of digital competition: Emerging organizational models. *Journal of Asynchronous Learning Networks*, 2(1), 66-95.

Janicki, T., Cummings, J., & Kline, D. M. (2014). Information technology job skill needs and implications for information technology course content. *Information Systems Education Journal*, 12(6), 59.

Jepsen, C., Troske, K., & Coomes, P. (2014). The Labor-market returns to community college degrees, diplomas, and certificates. *Journal of Labor Economics*, 32(1), 95-121.

Layman, L., Williams, L., Damian, D., & Bures, H. (2006). Essential communication practices for Extreme Programming in a global software development team. *Information and Software Technology*, 48(9), 781-794.

Liu, K. (2000). *Semiotics in Information Systems Engineering*, Cambridge University Press, Cambridge, U.K.

Longenecker, H. E., Feinstein, D. L., & Babb, J. S. (2013). Is there a need for a Computer Information Systems model curriculum? In Proceedings of the Information Systems Educators Conference, ISSN: 2167, 1435-1447.

Morey, A. I. (2004). Globalization and the emergence of for-profit higher education. *Higher Education*, 48(1), 131-150.

NCES. (2017). National Center for Education Statistics, retrieved July 12, 2017, nces.ed.gov/globallocator

Reynolds, J., Adams, R., Ferguson, R., & Leidig, P. (2017). Programming in the IS Curriculum: Are requirements changing for the right reason? *Information Systems Education Journal*, 15(1), 80.

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., Cross, J., Impagliazzo, J., LeBlanc, R., & Lunt, B. (2006). Computing Curricula 2005: The overview report. *ACM SIGCSE Bulletin*, 38(1), 456-457.

Solomon, N. A., Scherer, R. F., Oliveti, J. J., Mochel, L., & Bryant, M. (2017). The perfect match: Factors that characterize the AACSB International initial accreditation host school and mentor relationship. *Journal of Education for Business*, 92(3), 114-120.

Stamper, R. K. (1973). *Information in Business and Administrative Systems*. John Wiley and Sons, New York, NY.

Stamper, R. K., Althous, K., & Backhouse, J. (1988). MEASUR, Method for Eliciting, Analyzing, and Specifying User requirements. In Olle, T. W., Verrijn-Stuart, A. A. & Bhabuts, L., (eds.), *Computerized Assistance During the Information Life Cycle.* Elsevier, Amsterdam, The Netherlands.

SwitchUp.org. (2017a), retrieved July 7, 2017, www.switchup.org/... /are-coding-bootcamps-worth-it-job-placement-market

SwitchUp.org. (2017b), retrieved July 8, 2017, www.switchup.org/research/best-coding-bootcamps

Thompson, C. (2017, February). The next big blue-collar job is coding. *Wired Magazine*, 24(12).

Topi, H., Valacich, J., Wright, R. T., Kaiser, K. M., Nunamaker, J. F., Sipior, J. C., & 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 7, 2017, www.acm.org/education/curricula/IS%202010%20ACM%20final.pdf

USDoE. (2008). U.S. Department of Education, International Affairs Office, Structure of the U.S. education system: Credit systems, retrieved July 6, 2017, www2.ed.gov/about/offices/list/ous/international/usnei/us/credits.doc

Waguespack, L. (2011). Design, the “straw” missing from the “bricks” of IS curricula. *Information Systems Education Journal*, 9(2), 101.

West, D., Rostal, P., & Gabriel, R. P. (2005 October), “Apprenticeship agility in academia.” In Companion to 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, 371-374.

Wikipedia. (2017a) “Close Order Drill”, retrieved 7/6/2017, en.wikipedia.org/wiki/Military_parade

Wikipedia. (2017b) Coding Bootcamps, retrieved 7/7/2017, en.wikipedia.org/wiki/Coding_bootcamp

Worley, L. (2012). Outsourcing, Offshoring, Nearshoring, Onshoring – What's Going On? *Legal Information Management*, 12(1), 9-11.

Yourdon, E. (2002, February). Preparing software engineers for the 'Real World'. Proceedings of 15th Conference on Software Engineering Education and Training, 2002, CSEE&T, p. 3.

Yuan, L. & Powell, S. (2013). *MOOCSs and Open Education: Implications for Higher Education*, Centre for education technology and interoperability standards, retrieved 7 July 2017, publications.cetis.org.uk/wp-content/uploads/2013/03/MOOCs-and-Open-Education.pdf