

# Course, Program, and Curriculum Gaps: Assessing Curricula for Targeted Change

Barbara D. Gannod<sup>1</sup>, Gerald C. Gannod<sup>2</sup>, and Mark R. Henderson<sup>3</sup>

**Abstract** - In the Spring Semester of 2004, a gap analysis team at Arizona State University analyzed the impact of a significantly funded curriculum development program targeted at improving the abilities of interns and fresh graduates to develop embedded systems. To assist in this endeavor, the team identified three criteria for characterizing gaps between a desired and actual curriculum. In the past, the curriculum development program had been introducing new courses (targeting *Curriculum Gap*) in an attempt to update curriculum. However, due to program constraints, students were essentially unaffected by the curriculum changes (a *Program Gap*). It was determined that *Course Gaps* were the least problematic gaps to address in the short term and had a high impact on interns and graduates of the programs. This paper describes the set of criteria that we have developed for classifying different kinds of gaps and the process used in both gathering data and assessing a program to affect curriculum improvement. Specifically, we demonstrate that curriculum change requires a disciplined and systematic methodology that goes beyond simply adding new courses. Finally, we describe our experiences in applying gap classification and the impact of the gap analysis upon the curriculum program.

*Index Terms* – Accreditation, Assessment, ABET Constituents, Curriculum Reform

## INTRODUCTION

Changes made in 2000 by the Accreditation Board for Engineering and Technology (ABET), the major body for accrediting engineering and computing programs, introduced a shift in focus from assessment of curriculum *content* to assessment of program *outcomes*, primarily student outcomes. This shift in the target has turned out to be a huge adjustment for program administrators. What had been a *de rigueur* process for the past 50 years now presents a challenge of obtaining and evaluating information never before formally considered by most universities.

ABET specifies that a successful engineering program will produce students who can meet a set of defined requirements [1], including difficult-to-measure capabilities such as functioning on a team and interest in life-long learning. Now universities must measure “what is learned

instead of what is taught” [1], which results in several problems: 1) departments have not traditionally measured student outcomes so no process or data exists, 2) some of the criteria are difficult to define, let alone measure, and 3) it is not clear who should evaluate what students have learned: faculty, students, employers or someone else.

Universities can be viewed as producing a product: curriculum that prepares students that can contribute to their field of study. This product must undergo a continuous improvement process that involves assessment of the curriculum. *Verification and Validation* of products focuses on two primary questions: “*are we building the product correctly?*” (verification), and “*are we building the right product?*” (validation). From this perspective, many assessment techniques for measuring student outcomes (e.g., are students capable of demonstrating their capabilities in some area) are serving a curriculum verification role. Needs assessment, particularly identification of what the proper outcomes are for a degree program and its supporting curriculum, serves a validation role. In this paper, we are most interested in the latter (validation) as a driver for curriculum reform.

Olds et al. recently summarized current practices for assessment in engineering education in which they characterize methods as either *descriptive* or *experimental* [2]. In regards to verification and validation, the descriptive and experimental assessments identified can be considered to be verifications of whether current or proposed curricula meet program goals.

In the Spring Semester of 2004, a gap analysis team at Arizona State University (ASU) analyzed the impact of a significantly funded curriculum development program targeted at improving the abilities of interns and fresh graduates to develop embedded systems. In the context of verification and validation, the purpose of the analysis was to validate the program. That is, the purpose was to address the validation question: *are we delivering the right product (curriculum)*. To assist in this endeavor, the team identified three criteria for characterizing gaps between the desired and actual curriculum. In the past, the curriculum development program had been introducing new courses in an attempt to update curriculum. However, due to program constraints, students were essentially unaffected by the curriculum changes. It was determined that updating and modifying existing required courses were the least problematic changes in the short term

<sup>1</sup> Barbara D. Gannod, Assistant Professor, Division of Computing Studies, Arizona State University – East, bgannod@asu.edu

<sup>2</sup> Gerald C. Gannod, Assistant Professor, Division of Computing Studies, Arizona State University – East, gannod@asu.edu

<sup>3</sup> Mark R. Henderson, Professor, Dept. of Industrial Engineering, Arizona State University – Tempe, Mark.Henderson@asu.edu

and had a high impact on interns and graduates of the programs.

This paper describes the set of criteria that we have developed for classifying different kinds of gaps and the process used in both gathering data and assessing a program to affect curriculum improvement. Specifically, we demonstrate that curriculum change requires a disciplined and systematic methodology that goes beyond simply adding new courses. Finally, we describe our experiences in applying the gap taxonomy and the impact of the gap analysis upon the curriculum program.

The remainder of this paper is organized as follows. We begin by presenting background material in the area of assessment and describe the embedded systems program at ASU. We then introduce a taxonomy of gaps that we have identified for validation assessment of degree programs. Next we present an analysis of the embedded systems program at ASU using the gap taxonomy. Finally, we draw conclusions and suggest future investigations.

**BACKGROUND AND RELATED WORK**

In this section we provide background material in the area of assessment and describe the embedded systems program at Arizona State University.

*I. Assessment*

*Descriptive Studies* observe the current state of some phenomenon [2]. With respect to verification and validation, descriptive studies answer the question of whether the product is being developed correctly. In the context of curriculum assessment, descriptive studies serve to *verify* what students know and are able to do, or worded differently, verify whether course content is delivered correctly. Methods for performing descriptive studies include use of surveys, focus groups, and observation [2]. As an example, the Department of Computer Science and Engineering at ASU uses a pre-test/post-test survey given to students as one tool to assess student outcomes. The surveys allow faculty to determine the quality of courses with respect to current content.

*Experimental Studies* observe how changes to some artifact impact its overall performance [2]. In the context of curriculum assessment, experimental studies serve to *verify* the differences in the quality of what students know and are able to do as a result of the introduction of some perturbation. With respect to verification and validation, experimental studies answer the question of whether course outcomes are met with higher quality when some change is introduced.

In the approach described in this paper, we focus on the problem of *validation*. Specifically, we are interested in determining whether a program delivers the correct content (e.g., are we building the right product as opposed to are we delivering the product correctly).

*II. Embedded Systems at ASU*

In 2001, a consortium made up of ASU, Motorola, and Intel, known as the Consortium for Embedded Systems (CES), has combined to support the development of an embedded

systems concentration and embedded systems oriented Computer Systems Engineering (CSE) program [3]. As part of these efforts, the CES has funded a curriculum development program meant to affect the content of the CSE program at ASU. In addition, CES has funded a successful internship program intended to provide CSE students with experience developing embedded systems in an industrial environment.

The efforts described in this paper and elsewhere [4], were developed in order to *validate* the effectiveness of the CES curriculum program, which to date has funded fifteen projects targeted at developing new, or modernizing existing, course and laboratory content. While the assessment techniques have been created specifically to evaluate the CES funded program, they are applicable to any program where validation is of concern.

*III. Related Work*

Gerchak et al. describe their use of *concept maps* to measure integrated knowledge in order to assess students' understanding of their field of study [5]. The approach provides a technique for scoring concept maps using a holistic metric to evaluate programmatic objectives. From the perspective of verification and validation, this technique addresses verification in the sense that they are measuring student outcomes with respect to expected knowledge. Our approach, while less quantitative, is focused upon the validation of program objectives in order to determine whether the objectives are indeed appropriate for a given field of study.

Besterfield-Sacre et al. describe a framework for characterizing outcomes using Bloom's taxonomy of educational objectives [6]. The approach uses attribute lists to better characterize 11 specific but intentionally ambiguous ABET outcomes in order to adapt them into a given program. This approach, like ours, is intended for *validation* of a program. The primary difference, however, lies in the fact that our approach is meant to address topical outcomes rather than the more general ABET outcomes analyzed by Besterfield-Sacre et al.

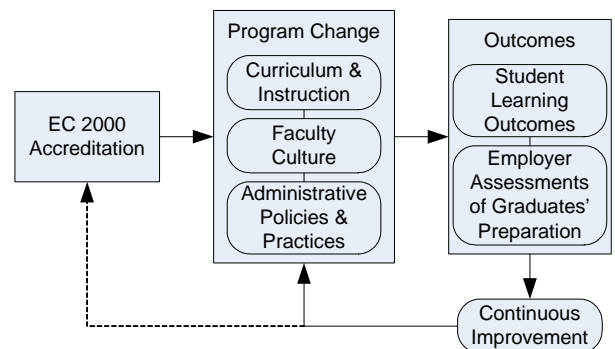


Figure 1 Conceptual Framework [7]

Prados et al. reported on the impact of Engineering Criteria 2000 (EC2000) as a means for assuring the quality of engineering education programs [7]. Figure 1 shows the conceptual framework that they introduced as means of

explanation of how assessment is to impact curricular change. In their work they surveyed 3,000 faculty, 185 program chairs, 11,500 graduating seniors, and 13,000 alumni on the state of the programs that they were associated with and found some interesting results. Specifically, they identified some of the sources of influence on curricular change as consisting of industry/employer feedback, ABET accreditation, student feedback, and an overall shift in department goals. While the conceptual framework they have developed accounts for change, there is no clear notion of a separation between *verification* and *validation*.

### TAXONOMY OF GAPS

In this section we provide a taxonomy of curriculum related gaps.

#### I. Overview

While assessment of student learning, a verification activity, is an important endeavor, part of the continuous quality improvement process is the assessment of curriculum itself. If students are studying the “wrong” curriculum, their mastery of the curriculum will have limited impact. Determining what the “right” curriculum is, a validation activity, is of utmost importance. In the Spring of 2004, a gap analysis team at Arizona State University performed a study of the Bachelor of Science in Engineering (BSE) in Computer Systems Engineering (CSE). The primary objectives of the gap analysis study were as follows:

- Determine what the desired curriculum should be,
- Compare the desired curriculum with existing curriculum and determine the “gaps” between the two, and
- Suggest changes that could be made to move the curriculum from its current state to the desired state.

The study revealed three different target areas for making changes: curriculum as a whole, individual courses, and programs themselves.

#### II. Curriculum Gaps

The first type of gap we identified, we call a *curriculum gap*. In particular, if a topic is determined to be necessary but does not exist in the current curriculum, that topic is identified as a curriculum gap. As a validation activity, we are asking: “are we offering the right curriculum?” Curriculum gaps expose the areas of the curriculum, as a whole, that are “wrong.”

When the current curriculum does not meet the needs of its constituents, perhaps the most obvious way to address the problem is to design new courses that provide coverage of crucial topics currently absent from the curriculum. For example, our study found that the use of scripting languages was considered a necessary skill for success in the development of embedded systems. Currently, the CSE curriculum does not provide coverage of scripting languages in any courses. To address this curriculum gap, a natural solution is to introduce a course that gives practical experience with several scripting languages.

#### III. Course Gaps

Sometimes the introduction of new courses is necessary to provide coverage of important concepts and skills. However, often times the material that is missing is related to courses that already exist in the curriculum. Because technology is so rapidly changing, courses can quickly become “out of date.” Thus, a second type of gap, what we call a *course gap*, exists. In particular, if courses in the current curriculum state that a desired topic is addressed, but constituents report knowledge in the area is not appropriate, the topic is a course gap. In terms of validation, we are asking: “is this the right course?” Course gaps expose topics in specific courses that are “wrong.”

Crucial topics in the desired curriculum can be introduced by replacing outdated material in existing courses and by providing greater depth of coverage of important topics that are currently covered “superficially” in a course. For example, our study showed that although all CSE majors are required to take a junior-level course in computer architecture, interns did not have the desired skills in the area of computer architecture. Upon further examination, we discovered that the gaps were in *modern* computer architectures and in depth of coverage. Introduction of a new course is unnecessary. Rather, making changes to the existing course will address the root of the problem.

#### IV. Program Gaps

Updating existing courses to include desired knowledge and skills, and introducing new courses that cover topics previously absent will have no effect if the constraints of the program do not allow students to be exposed to the courses in which changes occur. The third type of gap, which we term a *program gap*, addresses this issue. If courses in the current curriculum address a desired topic, but students cannot take the courses due to inflexibility in program constraints, the topic is identified as being a program gap. Here we are asking the validation question: “does our program include the right requirements?” Program gaps expose areas where the program itself is “wrong.”

The only courses students are guaranteed to take are those required by the program. Courses that are electives may not be taken, especially if the program has little or no room for electives. Many engineering programs are extremely prescriptive. They feel that in order to meet accreditation standards there is little room for flexibility. Consequently, if the program does not require desired knowledge and skills, it is very unlikely that students will be exposed to it. For example, computer networking was identified as a crucial topic for graduates of the CSE program. A general course in computer networks exists, and the CES funded development of a course on network processors. Neither of these courses is required. Students can take only two elective courses (6 credit hours), and students involved in the internship program typically use all elective credits for internships. Thus, many CSE students and virtually all interns do not take the networking courses. The only way to rectify this problem is to make changes to the program requirements themselves. This

could take on different forms such as freeing up more hours for technical electives or by making networks a required course.

Making room in an already full program for everything that is desired is extremely challenging. A program may need to make drastic changes in its requirements to eliminate the gaps between what *is* and what *should be*. Program gaps certainly present the hardest change to make.

**CURRICULUM ANALYSIS**

In this section we describe the process used to conduct the gap analysis, the types of information that were gathered for the analysis, and the results of the work that was done.

*I. Methodology*

The gap analysis committee conducted two 1.5 hour interview sessions with 5 - 6 member student groups. The procedure was to introduce a goal to the group and then gather information and comments on course topics and skills that were most useful for their internships. They were asked to categorize both technical and non-technical skills that were found useful and to identify specific courses where those skills were learned. They were also asked to identify skills that were not taught in courses that they believed were needed for their internships and to summarize the value of the internship upon their overall educational experience. In order to ensure equal participation by all students, the Affinity Process was used [4]. The Affinity Process (also known as the KJ method) [8] helps organize a large set of items into a smaller set of related items in an environment that allows all participants to feel free to contribute equally. The results consisted of several grouped post-it notes, each with a student-generated summary header card. The individual post-it note topics will be called *secondary items* and the header card summary topics will be called *primary topics*. Multi-voting was used to rank the resulting primary topics.

Two additional focus groups were convened, one exclusively with intern industry mentors and one exclusively CSE faculty. We invited all participants to identify curriculum outcomes for two groups of engineers: interns and new college graduates. Fifteen mentors and four faculty participated in these sessions. A mentor focus group was convened with 6 intern mentors from Intel and 9 from Motorola. Instead of the Affinity Process, we asked the mentors the following questions:

- What skills are needed for interns to be successful in their internship responsibilities?
- What skills are needed for new graduates?

Scribes recorded the mentor responses for the mentor focus group. The separate faculty focus group resulted in responses that turned out to be more general than those of the mentors and, therefore, were more useful in confirming the student and mentor concerns rather than itemizing them for a ranked analysis.

After the focus group meetings were completed, the assessment team performed an analysis to determine the gaps

between desired knowledge identified by students and mentors and the BSE degree offered by the Department of Computer Science and Engineering (CSE). The analysis was performed by creating cross-reference tables showing the relationships between student and mentor identified needs and courses in the CSE program. These tables enabled identification where gaps existed as well as those courses in the programs that contribute to embedded systems desired knowledge. Conversely, these tables also allowed the team to identify those aspects of the curriculum that did not contribute to embedded systems knowledge. Furthermore, by doing the gap analysis in this way, the team was able to identify the impact of current degree programs by examining the percentage of topics covered in core courses versus technical electives.

*II. Gap Analysis*

All courses in the CSE curriculum (both required courses and elective courses) were compared to the desired needs determined by the constituents. Tables 1 and 2 show examples of the data that was used to validate the appropriateness and impact of CSE courses.

Table 1: Student Needs Assessment

		Industry Mentor Concern	CSE Program
I. C/C++ and other non-Java languages (18)	a. Fundamentals of C/C ++ programming	✓	<b>240</b>
	b. Pointers in C/C++	✓	<b>240,</b> 494
	<b>c. Object Oriented Programming in C++</b>	✓	<b>240</b>
	<b>d. inheritance</b>	✓	<b>240</b>
	e. Advanced C/C++ programming	✓	494
	f. Pointers to functions (C/c++)	✓	494
	g. Understanding C code/low level use of a high	✓	
	h. Standard template library	✓	
	i. Writing to virtual/physical memory in C/C++	✓	494
	j. Visual Basic		
	k. Writing GUI		

The primary purpose of Table 1 is to compare the current content of courses to the topics identified by student interns as being “absent” from their classroom education but needed for success in their internship responsibilities. A secondary purpose is to compare the topics identified by students to the desires identified by industry mentors as a confirmation of their importance in the curriculum. The first column of Table 1 lists primary topics (header card) and their rank votes in parentheses. The second column of the table is the secondary items or skills that were identified by students as being both necessary and missing from the CSE program. The third column of this table indicates whether the secondary items were a concern also identified by industry mentors, while column four indicates courses in the CSE program whose course objectives state that they address the topic. Required courses are shown in boldface type (others are electives).

Tables similar to Table 1 were created for topics identified by industry mentors. We do not include them here for the sake of brevity.

Table 2 provides an alternate view. In particular, each of the stated objectives in a course is compared to the topics that constituents identified as being necessary. Columns one and two contain course descriptor and course topics, respectively. The course descriptor is simply the course number and name. Column three indicates whether a topic covered in a current course was found to be useful by students (e.g., assessment of the *as-is*), while Column four indicates whether the topic was identified as desired but missing from the program by the students, and column five indicates whether the topic was identified as necessary by industry mentors. The reference numbers in columns 4 and 5 (labeled “Student Gap Ref” and “Mentor Gap Ref”, respectively) refer to specific gaps from the respective “Student Needs” and “Mentor Needs” assessments tables contained in the gap analysis document. For instance, the tag “S.IV” refers to the student concerns (as denoted by the “S”) and the primary topic “General Networking”. The references in the “Mentor Gap Ref” column are numbered analogously with “j-k” referring to specific mentor identified subtopics.

Table 2: Coverage by Course Analysis

		Found Useful by Students	Student Gap Ref	Mentor Gap Ref
CSE 434 Computer Networks	Computer Networks and the Internet	✓	S.IV	M.iii.j-k
	Application Layer	✓	S.IV	M.iii.j-k
	Transport Layer	✓	S.IV	M.iii.j-k
	Network Layer	✓	S.IV	M.iii.j-k
	Link Layer	✓	S.IV	M.iii.j-k
	Network Security	✓	S.IV	M.iii.j-k

The example shown in Table 2 is interesting because it clearly reveals the existence of course and program gaps. In particular, the third column shows that students mentioned all topics covered in the course as being required for success during their internship experience. However, students also identified all topics in the course as being missing from the curriculum. Since there is a networking course, clearly the issue is not a curriculum gap. Either the content in the course is outdated and therefore not useful, or students are not exposed to the course (due to program constraints) and consequently the material is missing from their educational experience. In the case of the CSE program, the latter option seems to be the root of the problem.

III. Results

Analyzing the curriculum and course content in this way allowed us to target specific areas for change. In some cases topics were completely missing from the curriculum (i.e., curriculum gaps). However, in many cases gaps existed due to lack of modernization of course content or in depth of coverage of important topics (i.e., course gaps). For the CES members, the most revealing discovery was that many of the desired topics were covered, many of them in new courses funded by the consortium, but the students were not taking the courses due to program constraints (i.e., program gaps). In particular, CES funded the development of seven new undergraduate courses in CSE and modifications to three existing undergraduate courses (two of which are required). However, the target audience for those courses was unable to take them.

Table 3 shows the top ten desired topics, as identified by constituents, for the CSE curriculum and the gaps that were identified for these topics. Note that some of the topics in Table 3 are identified as having multiple gaps. For example, C/C++ (the most critical skill identified) has been identified as having all three types of gaps. Currently, the core course *CSE240: Introduction to Programming Languages* provides brief (approximately 5 weeks) coverage of C/C++ programming. Although this course is required by all students, the majority of student interns report that the course did not prepare them for their internship responsibilities. Thus, the topic is a course gap.

Table 3: Gaps in the CSE curriculum

Topic	Course Gap	Program Gap	Curriculum Gap
C/C++ programming	✓	✓	✓
Practical experience with the Linux Operating System		✓	✓
Operating Systems Internals		✓	
Device Drivers	✓		
General Networking		✓	
Applied Networking	✓	✓	
Advanced Hardware	✓		
Modern Systems Architecture	✓		
Scripting Languages		✓	✓
Embedded Systems Design and Development		✓	

Fortunately, CSE240 is not the only course with coverage of C/C++ programming. The projects in an elective course *CSE494: SW/HW Interface for Embedded Systems* are written in C/C++. Although CSE494 gives practical experience with C/C++ programming, very few if any BSE students take the course due to the limited number of technical elective credit hours in the program. Thus, coverage of C/C++ programming is also a Program Gap.

In order to adequately prepare students for careers in which the majority of their programming is done in C/C++ a few weeks of exposure and even one entire course may not be adequate. For example, students preparing for software careers (where Java may be the primary language in which they write programs) take at least 3 semesters of Java programming and likely program in Java in several advanced classes as well. Similar exposure to C/C++ seems logical to prepare students for Embedded Systems careers. Thus, C/C++ has been identified as a Curriculum Gap. Careful consideration of all options must be made in order to determine the best avenue to close the gap.

Table 3 reveals that the majority of the gaps lie in the course and program area. The CES curriculum program had been funding new course development, targeting curriculum gaps, but the addition of new courses was essentially ineffective because curriculum was not at the root of the problem. The committee found that changes must be made to modernize existing required courses, and that substantial changes to program requirements must be made to move the program from its current to the desired state.

After the Gap Analysis, which included feedback from all constituents, was completed the CSE department immediately reacted. During the Fall Semester of 2004, the BSE program was redesigned. Several course modifications have been proposed, and the program requirements have been drastically changed. The proposed curriculum is farther along the bureaucratic process than it has been in the past four years. Receiving input from all constituents that allowed the committee to clearly identify the root of the problems and specific methods for change enabled the department to target areas of change and make modifications that resulted in improved courses, curriculum, and program as a whole.

### CONCLUSIONS AND FUTURE INVESTIGATIONS

This paper presents a taxonomy of “gaps” between current and desired curriculum and describes how classifying gaps in this way allows departments to target specific areas of change toward the goal of continuous curriculum improvement. When assessing curriculum, departments must perform validation (“are we offering the right curriculum?”) and verification (“are we delivering the curriculum correctly?”) activities. The process described in this paper focuses on curriculum validation. If the curriculum is not right, how well we deliver it will be of little impact.

Based on our experiences in assessing the BSE degree in Computer Systems Engineering (CSE) at Arizona State University, we have identified a number of issues that have an impact on the process.

**Applicability.** A great number of factors need to be understood in order to determine how the process scales to different degree programs. Among the factors is a need to gain a better insight into the needs of different constituencies and how those needs translate into methods for data gathering

and analysis. Furthermore, the overall benefits and potential costs savings must be analyzed.

**Rigor and Repeatability.** Our experiences with the early stages of the process have led to the recognition that rigorous and repeatable methods are necessary to properly verify and validate degree programs. The methods need to be appropriate for the contexts and users that will be employing them and rigorous enough to facilitate detailed analysis in order to drive curriculum change.

**Automation.** Much of our experience described in this paper involved extremely user intensive data gathering and analysis. The approaches employed required on-site participation and extremely low-tech tools. It is our belief that in order for our approach to gain widespread acceptance, that opportunities for automation should be identified and utilized to their fullest potential. This is especially necessary for the data analysis portion of the approach where topics and gaps are cross-referenced across the curriculum.

### ACKNOWLEDGEMENTS

The authors would like to thank Scott Coleman, the director of the Consortium for Embedded Systems for his input and contributions to the 2004 gap analysis. We would also like to thank the students, industry mentors, and faculty that participated in the focus group meetings, without whom this work would not have been possible.

### REFERENCES

- Engineering Accreditation Commission, "Criteria for Accrediting Engineering Programs", *ABET Report E1 11/19/03*, November 2003.
- Olds, B. M., Moskal, B. M., and Miller, R. L., "Assessment in Engineering Education: Evolution, Approaches and Future Collaborations", *Journal of Engineering Education*, Vol. 94, No. 1, January 2005, pp 13-25.
- Huey, B., Prital, D., Dannenberg, D., and Robertson, J., "An Arizona Ecosystem for Embedded Systems," In *Proc. Of the 2001 IEEE International Performance, Computing, and Communications Conference*, April 2001, pp. 131-134.
- Gannod, B. D., Gannod, G. C., and Henderson, M. R., "Development and Utilization of a Processor for Incorporating Constituent Feedback Into Curriculum Improvement," In *Proceedings of the 2005 American Society for Engineering Education Annual Conference & Exposition*, June 2005.
- Gerchak, J., Besterfield-Sacre, M., Shuman, L. J., and Wolfe, H., "Using Concept Maps For Evaluating Program Objectives," *33rd ASEE/IEEE Frontiers in Education Conference*, November 2003, pp. 20-25.
- Besterfield-Sacre, M., Shuman, L. J., Wolfe, J., Atman, C. J., McGourty, J., Miller, R. L., Olds, B. M., and Rogers, G. M., "Defining the Outcomes: A Framework for EC-2000," *IEEE Transactions on Education*, Vol. 43, No. 2, May 2000, pp. 100-110.
- Prados, J.W., Peterson, G.D., and Lattuca, L.R., "Quality Assurance of Engineering Education through Accreditation: The Impact of Engineering Criteria 2000 and Its Global Influence," *Journal of Engineering Education*, Vol. 94, No. 1, January 2005, pp. 165-184.
- Kawakita, J. *The Original KJ Method*, Tokyo: Kawakita Research Institute, 1991.