Process Teaching and Learning in Engineering Education

Sheryl Duggins, Ph.D.
Associate Professor
School of Computing and Software Engineering
Southern Polytechnic State University
Marietta, GA 30060
sduggins@spsu.edu

Abstract

This paper explores the process-oriented constructivist theory of teaching and learning which has its roots in cognitive psychology, philosophy, learning theory, and education theory. A process model for teaching and learning software engineering is proposed and an initial set of maturity levels are defined. This process-based model for teaching and learning attempts to bridge the gap between constructivist theory and engineering education by graphically depicting the learning process from three evolving perspectives: the black box, the memory state, and clear box descriptions. The Maturity Process Teaching Model proposed here incorporates the ideas of constructivism, operational definitions, process-improvement, and Capability Maturity Model-based maturity levels and applies them to process teaching.

1. Introduction

One of the many challenges facing engineering educators is how to teach the subject matter in a way that both does justice to the material and stimulates the students to learn. This paper explores the process-oriented constructivist theory of teaching and learning which has its roots in cognitive psychology, philosophy, learning theory, and education theory. The constructivist approach to teaching focuses on active and cooperative learning in which the student is actively engaged in the learning process. In addition to referring to an educational philosophy, the term constructivism applies both to the underlying learning theory, which has its foundations in research in cognitive psychology, as well as to epistemology, or the nature of knowledge. This multi-disciplined approach suggests that the way educators teach should be based on how students learn, which should be directly related to the nature of knowledge and meaning.

A process-based maturity model for teaching engineering is proposed based on the constructivist theory of learning. The role of process as it relates to teaching and learning engineering is investigated and a maturity process teaching model is presented. To illustrate different levels of process understanding of student learning, box structures are utilized and examined from the perspective of the maturity teaching model.

2. Constructivist Theory

Constructivism is not a new idea. Students studying education theory, or learning theory, have been exposed to constructivist theory for decades. *Constructivism* stems from the work done by several theorists including Jean Piaget [1], Seymour Papert [2], Jerome Bruner [3,4], and John Dewey [5]. Education courses are frequently designed to be "how-to-teach" courses, but very few software engineering (SwE) or computer science (CS) professors, typically having been educated in Colleges of Engineering and not in Colleges of Education, have ever been exposed to this type of material. Specifically, very little has been written for the software engineering education community to address the specifics of *how* we should teach. We spend so much of our time focusing on *what* to teach, that we never seem to address *how* to teach it. Most of us still follow the traditional model of teaching in which the teacher, being the knowledge holder, lectures to the students, who are passively trying to receive and absorb as much knowledge from the teacher as they can. What is being suggested here is that we as a community, might benefit from exposure to research into how people learn, so that we might improve how we teach.

The basic ideas behind *constructivism* [3] focus on the students taking an active role in their own learning as they "construct" their own knowledge by integrating the new information with pre-existing semantic constructs. Key to this style of teaching and learning is the notion that the learning activity must be relevant and engaging to the student. This approach centers on problem-solving and critical thinking skills that the student utilizes by applying approaches based on their prior knowledge and experience to a new problem situation, and integrating those approaches with new experiences and knowledge, to construct a new level of understanding.

The teacher is typically viewed as a facilitator or coach in the *constructivist* learning approach. That means the teacher guides the student through the learning process by stimulating the student's critical thinking skills and providing learning situations, environments, skills, content, and tasks that are relevant and realistic and simulate real-world contexts. The emphasis is on active knowledge construction rather than just passive, knowledge reproduction. This knowledge construction is individual, but often occurs as a result of social experiences including negotiation, collaboration, and authentic experiences. The role of teacher as facilitator does not preclude the teacher from presenting new material in a formal class lecture; it just emphasizes the need to have the student actively involved in applying the knowledge in a problem-solving situation.

3. The Role of Process

One fundamental idea that is crucial to include in our software engineering curriculum is the role of process. We teach our students the importance of process in software engineering; however, we typically fail to incorporate the very fabric of process in the *way* we teach. The teacher as lecturer approach, which we frequently utilize, places the student in a passive role in the learning process. Constructivist theory tells us that learning is an active process in which the student

constructs knowledge. Learning is not the passive acceptance of knowledge, but rather that the learner needs to do something to learn.

Quality Assurance (QA) is based on the idea that improving the process by which a product is developed will result in an improved product [6]. The initial work in this area, known as Total Quality Management (TQM), was done by Deming [7] and was applied to the manufacturing community. Since his initial contribution, QA has been applied to all types of endeavors, with the underlying assumption that having a defined and managed process will improve the resulting product. With respect to software engineering, the Software Engineering Institute (SEI) has developed the Capability Maturity Model (CMM) to assess the quality of the software processes of an organization [9].

The concept of process, or specifically, software process, is fundamental to the notion of software quality assurance. Humphrey [11] defines the software process as "the sequence of steps required to develop or maintain software" and a software process definition as "a description of this process." The software process definition should guide the individual software engineers as they work. An organization that has a well-defined process description can better coordinate the work of individuals and track their progress. As new methods are identified, they are incorporated into the process definition, facilitating learning by allowing new projects to build on prior experiences.

The Capability Maturity Model provides a way for organizations to assess the capabilities of their current software processes and to focus on improving those processes. The CMM defines five levels of progressively more mature process capability [9].

"1. Initial: The software process is characterized as ad hoc and occasionally even chaotic. Few processes are defined, and success depends on individual effort.

2. Repeatable: Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier success on projects with similar applications.

3. Defined: The software process for both management and engineering activities is documented, standardized, and integrated into a standard software process for the organization. All projects use an approved, tailored version of the organization's standard software process for developing and maintaining software.

4. Managed: Detailed measures of software process and product quality are collected. Both the software process and products are quantitatively understood and controlled.

5. Optimizing: Continuous process improvement is enabled by qualitative feedback from the process and from piloting innovative ideas and technologies."

Most organizations are at level 1. The success of the CMM has focused attention on the successful role that a well-defined process can have on an organization. Organizations that effectively communicate and manage their processes operate more efficiently, and with experience, are more likely to improve those processes. In the terms of the CMM, those organizations mature, and reach a higher level of capability.

4. The Process Teaching and Learning Model

The role of process as discussed with respect to a software organization, and CMM in particular, presents a model of successively improving stages of maturity. One of the underlying assumptions of CMM is the idea of a defined process, or specifically, an operational definition, which Deming [7] refers to as something everyone in the organization can communicate about and work toward. Some of the benefits of using operational definitions [11] are that they enable effective communication, they enhance understanding, they facilitate reuse, and they support process evolution.

While CMM was originally intended to apply only to software organizations, this author is proposing that the ideas can be applied to other types of processes; namely, to teaching and learning processes. Using constructivist learning as the type of process-oriented learning that best fits into the software engineering curriculum, a new process-based model for teaching and learning software engineering is proposed. It is suggested that the five levels of the CMM could in a very broad sense, be applicable to the maturity of process teaching as well as to software organizations. Thus the five levels: 1. Initial; 2. Repeatable; 3. Defined; 4. Managed; and 5. Optimized, of CMM, could be redefined to apply to the maturity levels of process-based teaching demonstrated by an individual teacher in a particular course.

Paralleling Humphrey's ideas on software processes [11], a teacher who has a well-defined process approach to teaching will be able to assess and track the individual learning progress of her students. Furthermore, as the teacher matures and identifies new, successful, teaching and learning activities, they will be easily and effectively incorporated into her process teaching, which facilitates learning and improvement by the teacher, by creating new teaching activities that build on prior experiences.

The Maturity Process Teaching Model (MPTM) can be used to assess the process maturity level of individual teachers and to provide a technique for process-based improvement. Like CMM [9], the MPTM has five levels of increasingly more mature process teaching. The levels are as follows:

1. Initial: The teaching and learning process is characterized as typically ad hoc. Few teaching processes are defined, and learning depends on individual student effort.

2. Repeatable: Basic teaching processes are established and operational definitions are utilized. Process-based teaching is in place to repeat earlier success in previous classes having similar approaches and content.

3. Defined: Teaching and learning activities are documented, rehearsed, and integrated into a defined teaching process for the individual teacher. All courses taught by the teacher incorporate similar skills, strategies, and approaches, based on constructivist process-based teaching and learning, for creating environments that foster active knowledge construction.

4. Managed: Process teaching and learning are quantitatively assessed and evaluated.

5. Optimizing: Continuous teaching process improvement is enabled by frequent qualitative feedback assessing the process, and from incorporating innovative ideas and experiences in the process.

For decades, education literature has been promoting the use of operational definitions for evaluating teaching effectiveness. Also for decades, constructivist teaching and learning theories have purported that students learn more when they are actively engaged in constructing their own knowledge. CMM has shown that organizations that communicate effectively and utilize well-defined processes, are more productive. Fundamental to CMM and QA is the notion that process improvement based on experience is possible, and as organizations mature, they attain a higher level of capability. The Maturity Process Teaching Model proposed here incorporates the ideas of constructivism, operational definitions, process-improvement, and CMM-based maturity levels and applies them to process teaching.

While researching constructivism, the work of Harlan Mills [10] on the mathematical aspects of box structures came to mind. Box structures are the fundamental objects used for analysis and design of information systems in the box structure methodology. They are formally, (mathematically), defined, and I teach them in my graduate software engineering class on formal software specification methods. There is nothing about box structures that is inherently related to CMM, or process teaching or learning, or constructivism. But I found it useful to depict my process-based model using graphical structures that are similar to those used by Mills.

The process-based model for teaching and learning attempts to bridge the gap between constructivist theory and software engineering education. It graphically depicts the learning process of a student, from the three evolving perspectives: the black box, the memory state machine, and the clear box representations.

Initially, as we teach our students, we view each of them as a black box: we do not really know what is going on inside their heads. We do not know whether they are learning anything at all. All we know is that we are presenting them with various input stimuli (lecture, homework, textbook, learning activities, etc.), and we hope they are learning the content material. Thus we initially view the student learning process at a high level of abstraction, and at a low level of process understanding – as a black box. This perspective corresponds to level 1, or the initial maturity process level in the MPTM. See Figure 1.

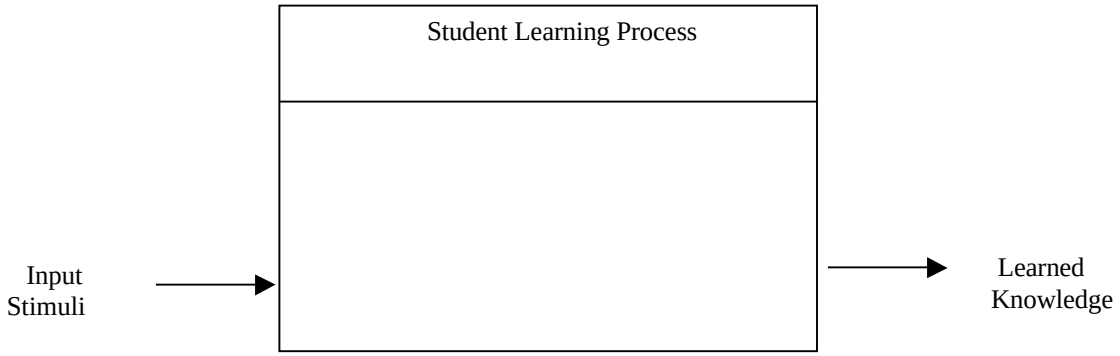

Figure 1.
A Black Box Description of Process Learning

The next level of process understanding and abstraction is based on the constructivist concept that learning occurs when new knowledge is "constructed" by integrating the new information with pre-existing semantic constructs. In this view, called the memory state machine description of learning, we know a little more about the learning process and we characterize the student learning process at a lower level of abstraction. Specifically, the memory state description focuses on the role of previously existing semantic constructs in the student learning process. See Figure 2.

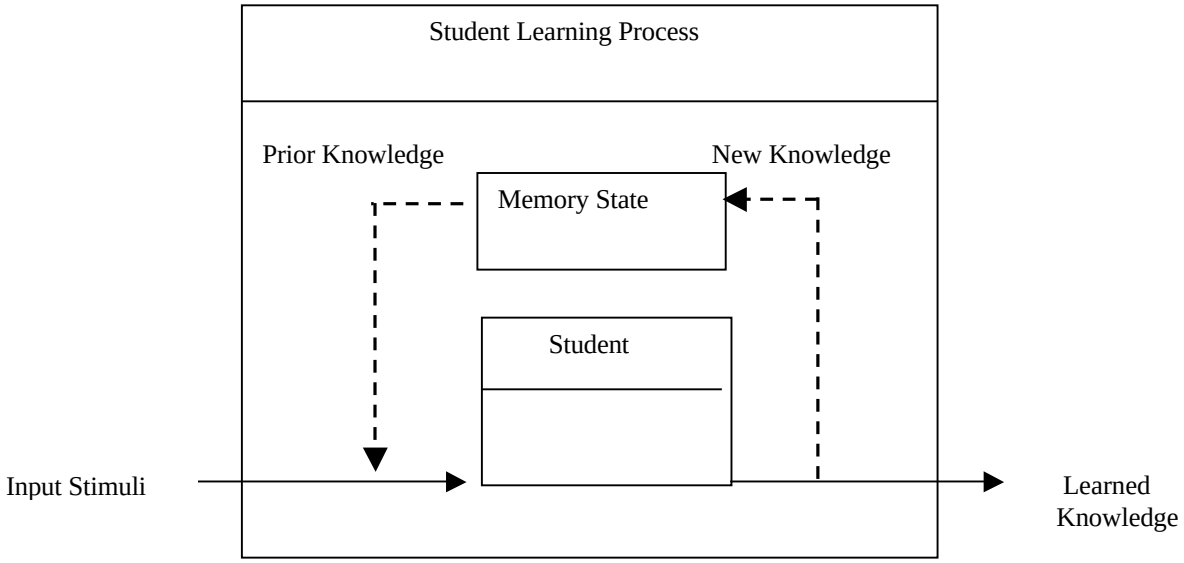

Figure 2.
A Memory State Description of Process Learning

The third level of maturity process understanding and abstraction examines the student learning process in more detail to determine whether information is being learned or not. Constructivist theory states that learning only occurs if the student finds the information relevant and the learning experience engaging. The alternation clear box description of process learning illustrates that learning principle, and also shows that information is forgotten, or not integrated with existing semantic constructs, and thus no new knowledge is constructed, when the student learner is not actively engaged in the learning process. See Figure 3.

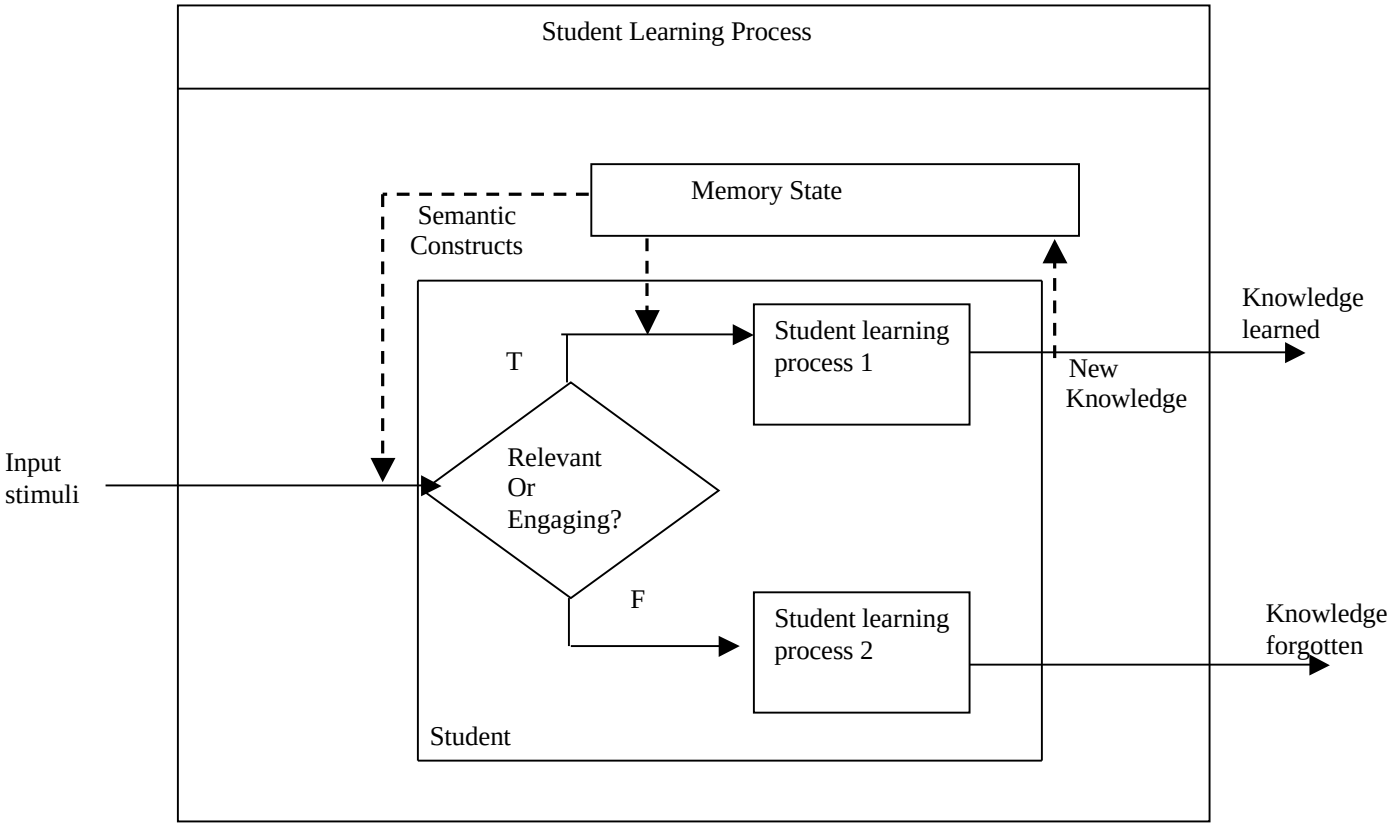

Figure 3.
Alternation Clear Box Description of Student Learning

Also at the third level of process understanding and abstraction is another clear box representation that examines how knowledge is learned. As the student constructs new knowledge, they apply approaches based on their prior knowledge and experience to the new problem situation, again integrating the new knowledge and experiences, to reach a level of understanding. I call this level the concurrent clear box description of process learning, because

the integration of the prior knowledge and experience occurs simultaneously and instantaneously. See Figure 4.

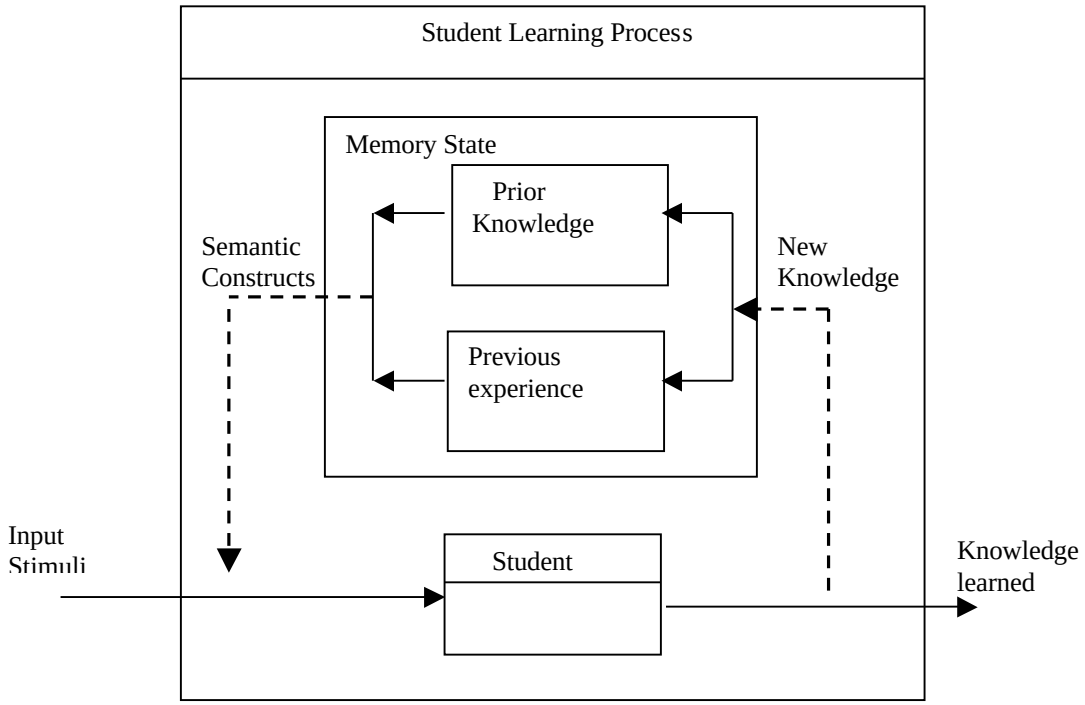

Figure 4.
Concurrent Clear Box Description of Memory State

These box diagrams depict progressively more detailed descriptions of the student learning process. Note that all four diagrams: black box, memory state, alternation clear box, and concurrent clear box, display the same student learning process, but at increasing levels of detail. There is an "includes" relationship between the boxes; the clear box descriptions are said to be included in the memory state description; the memory state description and the clear box descriptions are said to be included in the black box description. One can think of "opening up" or understanding more about the prior level as one progresses to a more detailed, mature, description of the student learning process.

As the level of maturity increases from one to five in the MPTM, the teaching process becomes more effective. The teacher is better able to create learning situations that map onto the more detailed views of the student learning process, and is able to do this in a defined, repeatable, and measurable way.

5. Future Work and Summary

This paper represents the initial stage of this investigation. The next stage of the study will map specific learning objectives to the maturity levels by applying constructivist theory to specific learning outcomes in software engineering. The final stage of the study will result in assessment techniques to evaluate the maturity level of software engineering educators based on the MPTM, and teaching processes to incorporate to facilitate maturity evolution of process teaching.

In summary, an initial process-based model of teaching and learning was proposed based on the constructivist theory of learning. The role of process in Quality Assurance as it relates to software engineering, and the process-based Capability Maturity Model were examined and applied to the process-based teaching model described. The Maturity Process Teaching Model was presented, and the five levels of maturity were defined. Three types of box structures: black box, memory state, and clear box, were created to illustrate different levels of process understanding of student learning, and they were examined from the perspective of the MPTM.

References

[1] Piaget, Jean (1966) The Psychology of Intelligence, Littlefield, Adams, Totowa, NJ.
[2] Papert, Seymour (1991) Constructionism: Research Reports and Essays, 1985-1990, Ablex.
[3] Bruner, Jerome S. (1960) The Process of Education, Harvard University Press, Cambridge, MA.
[4] Bruner, J. (1966) Toward a Theory of Instruction, The Belknap Press of Harvard University Press, Cambridge, MA.
[5] Dewey, J. (1994) "Thinking Technology", in Educational Technology, 34(4), 34-37.
[6] Wheeler, S. & Duggins, S. (1998) "Improving Software Quality", in Proceedings of the Southeastern ACM Conference, Marietta, GA, April 1998, pp. 300-309, ACM, New York, NY.
[7] Deming, W. E. (1986) Out of the Crisis, M.I.T., Center for Advanced Engineering Studies, Cambridge, MA.
[8] Vygotsky, L. S. (1978) Mind in Society: The Development of Higher Psychological Processes, Harvard University Press, Cambridge, MA.
[9] Paulk, Mark C. et.al. (1993) "Capability Maturity Model for Software, Version 1.1", Software Engineering Institute Technical Report, CMU/SEI-93-TR, February 1993.
[10] Mills, Harlin, Linger, Richard, & Hevner, Alan R. (1987) "Box-Structured Information Systems", IBM Systems Journal, 26(4), 395-413.
[11] Humphrey, Watts (1995) A Discipline for Software Engineering, Addison-Wesley.

Sheryl L. Duggins

Sheryl Duggins is an associate professor in the School of Computing and Software Engineering of Southern Polytechnic State University in Marietta, Georgia. She has served as the Graduate Coordinator for Software Engineering and was integral in developing the Master of Science in Software Engineering degree. She obtained her Ph.D. degree in Computer Science from the University of Florida in 1991.