2010 23rd IEEE Conference on Software Engineering Education and Training

An Undergraduate Summer Research Program in Software Safety[*]

W. Eric Wong and Vidroha Debroy
Department of Computer Science
University of Texas at Dallas
{ewong,vxd024000}@utdallas.edu

Abstract

This paper shares the experiences and lessons learned from conducting an NSF-sponsored eight-week summer research program for ten undergraduate students from multiple universities. The focus of the research was on "*Verification and Validation for Software Safety*", and emphasized a strong fundamental knowledge of software safety while maintaining a close collaboration with industry. The program included lectures and special tutorials from guest speakers, field trips, review sessions, as well as posters and final presentations by the students.

1. Introduction

Recent years have seen the development of Software Engineering into a discipline of its own [8,9], as opposed to just being used synonymously with Computer Science. In fact more and more universities now offer degrees in software engineering both at the undergraduate and graduate level. Some of these software engineering programs, and their constituent courses, have been developed based on the guidelines provided by the Software Engineering Institute (SEI) at Carnegie Mellon University. However, despite being recommended as an independent curriculum module [2,4] software safety (defined as freedom from software-enabled accidents or losses) fails to receive as much attention as it arguably should in software engineering curricula.

The topic of software safety has always been considered important for areas such as defense, nuclear-energy research, and aeronautics. Recently it has become even more important for software engineers to study this topic as computers are increasingly used to monitor and control safety-critical devices and processes, in areas such as medicine, transportation, etc. [4]. That being said, instead of teaching software engineers about software safety once they have already achieved professional status, it makes more sense for the principles of software safety to be built into the very fundamentals of a software engineer's body of knowledge, i.e., while the software engineer is *in becoming*. Recognizing this need, new programs such as the one in [1] have been proposed that aim to add a safety and/or mission critical track as an addition to the Software Engineering Master's program. An obvious problem of this approach is that a software engineer's education does not begin at the Master's level and in fact, some students enrolled in Master's programs are already software practitioners. Therefore, active steps need to be taken to ensure that knowledge from software safety, and its related areas, is imparted to students at the undergraduate level itself.

We decided to take one such step in this direction by hosting an NSF-sponsored summer REU (Research Experiences for Undergraduate) program at the University of Texas at Dallas (UTD) that focused on "*Verification and Validation for Software Safety*." Not only would undergraduate students be introduced to software safety, but it also meant that they would be exposed to conducting research at a relatively earlier level in their academic life. This holds additional benefits in that: first, the quality of the students' work is expected to increase via rigorous training; and second, research interests are to be stimulated which would possibly provide incentive to the students to pursue higher education. Thus, this paper serves to document and share our experiences in conducting such a summer research program, as well as report on the lessons learned as the program progressed.

[*]This work is sponsored by NSF (CCF-0851848). Professor W. Eric Wong is the faculty supervisor and the principal investigator of this research initiative. Mr. Vidroha Debroy is a PhD student as the student mentor.

1093-0175/10 $26.00 © 2010 IEEE
DOI 10.1109/CSEET.2010.36

The remainder of this paper is organized as follows: Section 2 gives an overview of our REU program and provides information on the students selected. Section 3 describes the research projects that the students worked on. Section 4 discusses how the program was conducted and some of its activities. Subsequently, the lessons learned are presented in Section 5, and the conclusions and future direction in Section 6.

2. The REU Program

The Research Experiences for Undergraduates (REU) program supports active research participation by undergraduate students in any areas of research funded by the National Science Foundation (NSF). An REU site consists of a group of undergraduate students who work with faculty and other researches at a host institution on specific research projects [7]. Different REU sites therefore have different projects and as mentioned before, the primary focus for our REU program at UTD is "*Verification and Validation for Software Safety*".

2.1 The Summer 2009 Program at UTD

An eight-week research program from June 1 to July 24 was offered, which received additional support from both the Computer Science Department and the School of Engineering to ensure its success. A detailed description of all the activities appears in Section 4. Below, we focus on two unique features of the program.

The most important uniqueness was that a very close collaboration with the industry partners was provided. In addition to work on assigned research projects at UTD, special field trips were arranged by members of the Industry Advisory Board to Raytheon, Lockheed Martin Aeronautics Company, and EDS/HP to help the students better understand how software safety is verified and validated in practice for real-life applications. This also gave students a chance to directly communicate with practitioners to receive a first-hand account of the work environments and lifestyles in the industry. These encounters would hopefully encourage the students to pursue a future career in a science or technology-related field.

Another special uniqueness was the opportunity for students to work with researchers and engineers from an NSF Net-Centric Software & Systems Industry/University Cooperative Research Center (Net-Centric IUCRC) [6] located in the Dallas area. The center is made up of three universities: UTD, UTA (University of Texas at Arlington), and UNT (University of North Texas), and twelve industrial partners including Raytheon, Lockheed Martin Aeronautics Company, EDS/HP, Boeing, Texas Instruments, and Rockwell Collins. It provided a rich environment for collaboration between industry and academia. Bi-weekly meetings were held among faculty, researchers, and engineers from member universities and industry affiliations. REU students were invited to participate in the activities such as four seminars sponsored by industry members of the center, while attending the summer program. These seminars ranged from "*Introduction to Software Safety*" to a more advanced special topic on "*Software Safety Applications in the Aeronautics Industry*". Specially prepared in-class and take home exercises were given to help students achieve a comprehensive understanding of the materials discussed at these seminars. Follow-up meetings were also organized for students to discuss with invited speakers how to address the gap between the current "state-of-practice" and "state-of-art" with respect to software safety.

2.2 The Students

The REU program was publicized through extensive distribution of brochures and invitations as well as a dedicated REU website (http://paris.utdallas.edu/reu) hosted at UTD. Our campaign was especially targeted at traditional minority institutions such as those with historically large African-American, Hispanic, Native American, and female enrollment. A list of these institutions can be obtained from sources such as *Project Kaleidoscope*, one of the leading advocates in the United States for "*what works*" in building and sustaining strong undergraduate programs in the fields of science, technology, engineering and mathematics (STEM) [3,5]. To encourage the participation of female undergraduate students, a special emphasis was put on women's colleges [10] both locally and nationally – especially those in NSF EPSCoR (Experimental Program to Stimulate Competitive Research) states. Program announcements, flyers, and related information were mailed to the administrative staff at selected institutions.

Students were invited to submit their applications which were evaluated on an equal opportunity basis. Those from underrepresented groups (women, minorities and persons with disabilities) and academic institutions with limited research opportunities were especially welcome. The selection was conducted to create a broad pool of participants and also ensure that everyone has a positive learning experience. A strong academic record and good potential for growth was critical. Some selection criteria are listed below:

- **Major**: Students majoring in Computer Science or Software Engineering were preferred.
- **GPA**: A GPA of over 3.0 was preferred.
- **Academic Year**: Incoming seniors were preferred.
- **Knowledge/Skills**: Knowledge in fundamentals of software engineering, safety analysis (e.g., FMEA: Failure Mode and Effects Analysis, FTA: Fault Tree Analysis), UML (The Unified Modeling Language), program verification, and/or software testing was preferred.

One point which we would like to emphasize is that an opportunity was provided to students who had shown strong motivation in science or engineering but were unable to adequately explore their interests due to limited resources at their home institutions. An example is that a student was accepted, who although had a good GPA and strong recommendations from his professors, but did not have the exact background and qualifications required for our research projects. To solve this problem, we provided a series of training seminars which would bring all recruited students up to speed and helped them to complete their research assignments.

Based on the above selection criteria, a total of 10 students were selected from more than 50 applicants. Of them, five were from UTD and the others were from different universities. With respect to the gender, three were female and seven were male. Each student received a stipend, housing allowance, and a travel imbursement for those who were not local residents. A correspondence was set up with students prior to their arrival to accommodate those who might have had special needs. On-campus housing was allocated to students who were unable to arrange for other living arrangements.

While several of the students had already taken at least an introductory course in software engineering prior to the program, none of them had had any experience with software safety, or had taken a related course before. Furthermore, only 2 out of the 10 students had ever been involved in active research and therefore, the choice of a software safety-based research project allowed us to address both of these inadequacies. All but one of the students were already in their senior year and the remaining student was one semester shy of reaching senior status. Thus, none of the students required very basic or elementary training in programming and software engineering practices.

3. The Projects

After deliberation and careful analysis, two umbrella research areas for the students to focus on were selected by the principal investigator. These were – "*Foundational Requirements for Competency in Software Safety*" and "*Testing for Software Safety*". Within each research focus, projects were designed such that they presented a reasonable workload for the students, yet at the same time allowed them to learn as much as possible in the short eight-week duration of the summer program. The first focus area was explored by means of two projects while the second area via a single project. Thus, a total of three projects were selected for the students to work on. Each of the project titles along with the project's designated research focus is provided in Table 1.

Table 1: Projects completed during the summer program and their corresponding research focus

Research Focus	Project
Foundational Requirements for Competency in Software Safety	I: An Evaluation of Software Standards
	II: The Role of Software in Catastrophic Accidents
Testing for Software Safety	III: Software Safety Analysis- An Integrative Approach

The students were divided up into groups of 5 such that each group was to work on one area of focus. Thus, projects I and II in Table 1 were covered by one group of students, and project III by another. A brief description of each of the projects is given below.

3.1 Project I: An Evaluation of Software Standards

In order to develop safe software, engineers and programmers must apply standards to software development processes and closely adhere to these standards in order to have a degree of confidence and trust in the process. However, today there exist hundreds of overlapping software safety standards that interact with an even larger number of additional standards and frameworks. So the fundamental question is:

How does a safety engineer select a safety standard well suited to his/her project?

In order to help make such a choice, this project aimed to perform an evaluation of safety standards across several proposed criteria. The students were expected to contribute in the following ways:

a) Proposing new criteria, and selecting from existing criteria to evaluate and compare software safety standards.
b) Selecting software safety standards for further analysis and evaluation against the criteria identified at Step (a).

3.2 Project II: The Role of Software in Catastrophic Accidents

Software today is being used to control an increasing number of safety-critical systems such as medical devices, nuclear power generators, cruise missiles, space shuttles, aircrafts, etc. However, in the past, unsafe practices in the implementation of these software systems have partly led to costly disasters and even the loss of life. The goal of this project was for the students to study these accidents and partially (if not completely) identify their root causes and the lessons that can be learned from them so that such mistakes can be avoided in the future. Thus, the students were expected to contribute in the following ways:

a) Identifying relevant catastrophic accidents suitable for study where software was cited as one of the significant causative factors of the accident.
b) Reviewing the contributing factors that led to each accident and dividing them up into those that were software-related and those that were not.
c) For the contributing factors that were software-related, examining why the software did not function correctly and how this might be avoided in the future.

3.3 Project III: Software Safety – An Integrative Approach

Software safety analysis plays an essential role in the development of any software-dependent mission- and safety-critical system. However, the safety analysis process used today is not based on an integrated model considering both functional and safety specifications simultaneously. It does not give a thorough analysis of all possible failures. The objective of this research is to integrate safety analysis methods with standard functional requirements to reduce failures inherent in performing the two independently. This can be accomplished by integrating the system's functional behavioral model (represented by UML state machines) with fault tree diagrams derived from the safety hazard analysis, which will give a clearer picture of the possible system failures. The identified system failures can be used in guiding subsequent validation to ensure the safe operation of the system. The students were expected to work on this integrative approach and contributed in three ways:

a) Identifying data and control dependencies between safety requirements and functional specifications
b) Developing an algorithm to support the integration of fault tree models and UML behavior state machines
c) Performing case studies to validate the integrated model.

4 Conducting the Program

In this section we describe how the REU program was conducted and in doing so we split the program into three phases: "The Beginning," "The Middle," and "The End". This division is only used for descriptive purposes and the phases may overlap one another slightly.

4.1 Program Kickoff: The Beginning

The start of the program was marked by a kick-off meeting that involved all of the students coming together for the first time and introducing themselves to one another. They were also introduced by the principal investigator (who had already met with the students individually before) to student mentors. Additionally, the Dean of School of Engineering and the Chair of Computer Science Department were also present at the meeting. The students were briefed on the projects that were to be undertaken, the resources that would be made available, and an overview of UTD.

The first week involved dividing the students up into two groups based on their relative interest levels in each project and preparing a working schedule for each group. Training courses and tutorials were also given to those who had not been exposed to topics such as software safety and UML before. Special steps were taken to ensure that each group had a relatively equitable balance of skills. The students were provided with a Lab where each student had internet access via their own computer. They were also provided with free access to printers and were assigned a netid (valid only for the duration of the summer program) so that non-UTD students could also access some of the other resources that would have otherwise only been available to the UTD students.

4.2 Work and Play: The Middle

The middle of the program (which was generally weeks 2 through 7 of the eight week program) was when the bulk of the research was done by the students. This period involved regular meetings between the principal investigator, student mentors, and the groups. One of the members from each group was assigned as a group leader and was responsible for submitting drafts of the various deliverables to the principal investigator on a regular basis. Each of these drafts was regularly reviewed and feedback was provided to the groups.

Aside from the assigned meeting times, the students were free to come and go from their Lab as they wished. The intention behind this was to provide the students with an atmosphere where they did not feel pressured but rather were given the opportunity to be creative. However, the students were asked to adhere to the deadlines as best as possible and when there were problems meeting the schedule, suitable alterations were made in order to meet the remaining deadlines on the schedule. The students were also told to contact the principal investigator or student mentors at the earliest should they have any problems, or need any guidance. Fortunately enough, there were no significant issues, nor was there any dissent or internal conflict between groups or between the members in any group.

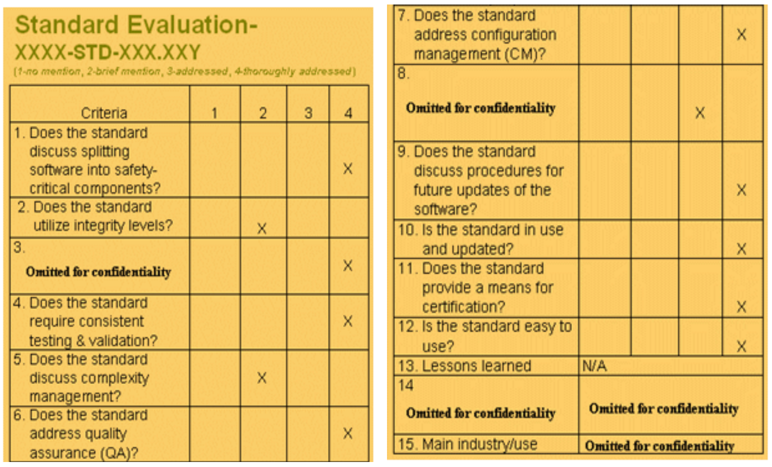
We also wanted to emphasize a strong relationship between academia and industry during the course of the program. Therefore, efforts were made to invite guest lecturers from the industry partners so that the students could contrast the knowledge taught to them at school with that which was applied in industry, as well as benefit from the experiences shared by the practitioners. For example, students attended a session where they learned first-hand about some of the riskmanagement practices in industry and were also asked to participate in classroom and take home exercises and presented their solutions to guest lecturers during the subsequent meeting for more in-depth discussions.

In addition to the research that was assigned to the students, we also wished for them to have an enjoyable experience over the summer and especially for those from other universities to travel around a bit. Thus, we organized field trips such as the one where students were taken on-site to Lockheed Martin Aeronautics Company for a tour that walked them through the assembly lines of F16, F22, and F35, and were allowed to interact first hand with key personnel and discussed the importance of software safety in their practice. A 4th of July barbeque party was also planned where students were to invite their friends and take a break from research. Also in attendance were the Chair and Associate Chair of Computer Science Department and they provided valuable insights to the students regarding their education.

4.3 Presenting Their Research: The End

Towards the end of the program (during the 7th and 8th week) the students finalized their deliverables and prepared three posters – one for each project. The summer program culminated in a meeting where the students were allowed to present the work that they had done over the course of the program, and were also to host a poster session. The meeting was attended by representatives from all of the industry partners, as well as faculty and school administrators at UTD. The audience members were allowed to ask questions regarding the presentations and posters, and the students were given the opportunity to defend their research work. Based on feedback from faculty members and industry affiliates, the students did an excellent job and exceeded expectations.

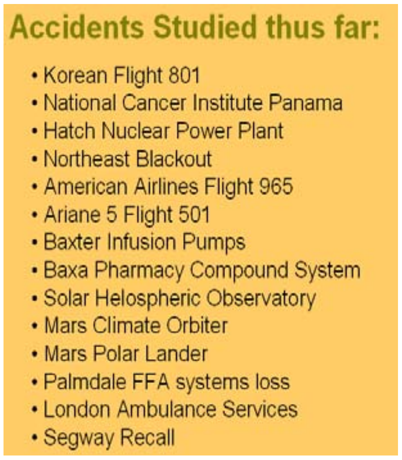
Due to confidentiality and space constraints, a detailed description of the work that was presented is not provided in this paper. However, a partial snapshot of material included in the posters has been given as an example in Figures 1 to 3. The final presentations and poster sessions were held on the last day of the eighth week and therefore, this marked the official end of the REU summer program.



Standard Evaluation-
XXXX-STD-XXX.XXY
(1-no mention, 2-brief mention, 3-addressed, 4-thoroughly addressed)

Criteria	1	2	3	4
1. Does the standard discuss splitting software into safety-critical components?				X
2. Does the standard utilize integrity levels?		X		
3. **Omitted for confidentiality**				X
4. Does the standard require consistent testing & validation?				X
5. Does the standard discuss complexity management?		X		
6. Does the standard address quality assurance (QA)?				X
7. Does the standard address configuration management (CM)?				X
8. **Omitted for confidentiality**			X	
9. Does the standard discuss procedures for future updates of the software?				X
10. Is the standard in use and updated?				X
11. Does the standard provide a means for certification?				X
12. Is the standard easy to use?				X
13. Lessons learned	N/A			
14 **Omitted for confidentiality**	**Omitted for confidentiality**			
15. Main industry/use	**Omitted for confidentiality**			

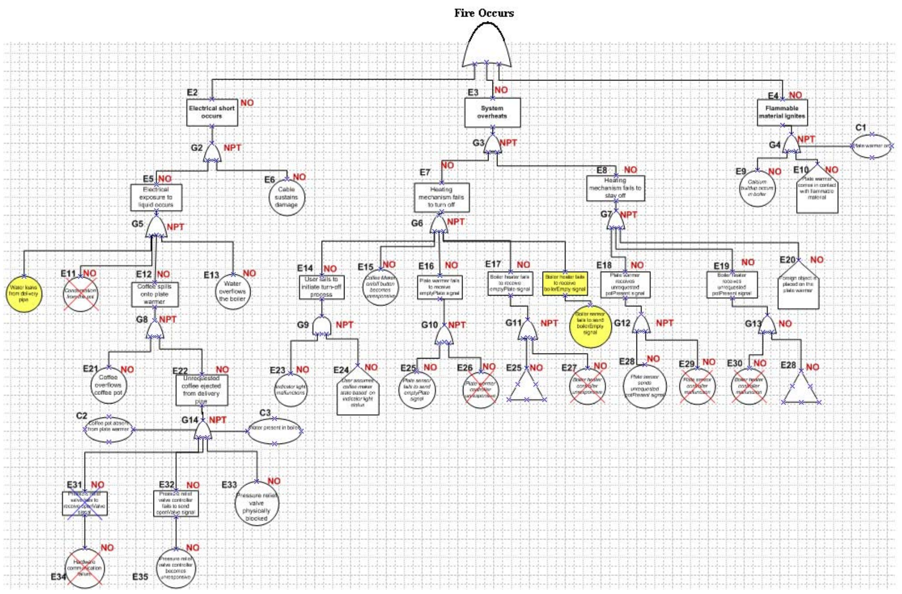
Figure 1: A standard evaluated by the students for Project I



Accidents Studied thus far:

- Korean Flight 801
- National Cancer Institute Panama
- Hatch Nuclear Power Plant
- Northeast Blackout
- American Airlines Flight 965
- Ariane 5 Flight 501
- Baxter Infusion Pumps
- Baxa Pharmacy Compound System
- Solar Helospheric Observatory
- Mars Climate Orbiter
- Mars Polar Lander
- Palmdale FFA systems loss
- London Ambulance Services
- Segway Recall

Figure 2: Some accidents studied by the students for Project II


Fire Occurs
E2
Electrical short occurs
NO
E3
System overheats
E4
Flammable material ignites
G2
NPT
G3
G4
C1
E5
Electrical exposure to liquid occurs
E6
Cable sustains damage
E7
Heating mechanism fails to turn off
E8
Heating mechanism fails to stay off
E9
E10
G5
G6
G7
E11
E12
Coffee spills onto plate warmer
E13
Water overflows the boiler
E14
User fails to initiate turn-off process
E15
E16
E17
E18
E19
E20
G8
G9
G10
G11
G12
G13
E21
Coffee overflows coffee pot
E22
Unrequested coffee ejected from delivery pipe
E23
E24
E25
E26
E27
E28
E29
E30
C2
C3
G14
E31
E32
E33
Pressure relief valve physically blocked
E34
E35
Figure 3: A Fault tree to illustrate how fire may occur in an electrical coffee maker for Project III

5 Lessons Learned

This was our first time hosting such an event at UTD and therefore, there are several important pearls of wisdom that we have taken away from this experience. In this section we outline some of the significant lessons learned and hope that others too might benefit from such experiences when taking on similar endeavors.

a) ***Enforcing an acceptable level of communication skills is very important***. We encountered one student whose level of English was sub-par and this presented several problems for the program. First, the student was having trouble keeping up with the other members in the group, and second, the student was having trouble successfully communicating his ideas and input to the other members of his group. This problem was alleviated thanks to another team member who spoke the same native language and thus, was able to act as an intermediary.

b) ***Undergraduate students may be capable of writing, but not always technical writing***. Since for most of the students, this was the first time they were working on a research project, they had not previously been exposed to much technical writing. As a result, some of the students needed to be taught how to correctly cite references, etc. This is highly surprising as all of the students had previously taken a "*Professional Communication and Technical Writing*" course (or one similar to it), and suggests that the quality of such courses may need to be improved.

c) ***Undergraduate students are capable of doing good research***. Prior to the launch of the program, we were skeptical and unsure as to how the students would perform when working on research projects of such importance. As a result we grossly underestimated the quality of work expected from them. We now stand corrected, and encourage students everywhere to pursue research projects from their undergraduate years themselves, and urge faculty members to make such projects available to their students.

d) Perhaps the most important lesson that we have learned is that safety ***is a very important subject of study for students***, and they should be given the opportunity to study it while they are still undergraduates. Such a claim is ratified by our own faculty and industry affiliates, which is why we are currently drawing up plans for a course on *software safety* to be offered to undergraduate students at UTD.

6 Conclusions & Future Direction

An eight week summer research program for undergraduates sponsored by the National Science Foundation was hosted at UTD. The research program focused on "*Verification and Validation for Software Safety*" and encompassed three different projects that were worked on by ten undergraduate students. A strong partnership with industry was emphasized and the students were exposed to guest lectures from our industry partners as well as field trips. The entire program was received very well by students, faculty members and industry partners alike.

Significant positive outcomes of the program have already begun to be observed. Six of the 10 students who participated have applied to graduate schools to pursue their advanced degrees. Four students continued to work on research projects related to software safety at UTD, and one of them is a recipient of an Undergraduate Research Award from the Office of the Vice President for Research at UTD. Two research papers have also been produced to illustrate the significance of the research results obtained, of which one has been submitted to a premier conference on software reliability, and the other to a premier software engineering journal. A course on Software Safety for undergraduates at UTD is also in preparation.

In subsequent instances of the REU program over the remaining two project years (2010 & 2011), we will make continuous improvement based on the lessons we have learned to ensure students who attend our program enjoy a more exciting research experience. Our future direction is to make this REU program an opportunity for students to gain proficiency in a broadly applicable skill set including oral and written communication, research methods, critical thinking, and problem solving. Therefore, regardless of whether they continue in computer science or software engineering, students will have acquired valuable skills that will better prepare them for their future field of study. In particular, their exposure to research may motivate them to continue on to graduate school and a career in science and technology.

Acknowledgements

We wish to thank the National Science Foundation as well as the School of Engineering and the Computer Science Department at the University of Texas at Dallas for giving us the opportunity to host such a program. We also thank our industry partners whose valuable inputs and support allowed the students to maximize their knowledge. And, last but certainly not least, we are extremely grateful to the ten undergraduate students whose constant enthusiasm and eagerness to excel made this program a wonderful, yet learning, experience for us as well.

References

1. C. Atkinson, D. Eichmann and C. McKay, "An evolution of a software engineering curriculum", in *Proceedings of the 8th SEI Conference on Software Engineering Education*, pp. 99-112, New Orleans, Louisiana, USA, March 1995
2. D. Budgen and J. E. Toamyko, "The SEI curriculum modules and their influence: Norm Giibs' legacy to software engineering education", *Journal of Systems and Software* 75(2005): 55-62, July 2004
3. Project Kaleidoscope (http://www.pkal.org/collections/WhatWorks.cfm)
4. N. G Leveson, "Software Safety," *SEI Curriculum Module SEI-CM-6-1.1*
5. J. L. Narum, "What Works: Building National Science Communities, Resources for Reform, Strengthening Undergraduate Science and Mathematics," *A Report of Project Kaleidoscope*, Volume II, Appendix D, 1992.
6. Net-Centric Software & Systems Center (http://netcentric.cse.unt.edu/)
7. NSF REU Program (http://www.nsf.gov/crssprgm/reu/)
8. M. J. Sebern, M. J. Lutz, M.J, "Developing undergraduate software engineering programs," in *Proceedings of the 13th Conference on Software Engineering Education & Training*, pp. 305- 306, Austin, Texas, March 2000
9. T. J. Reichlmay, "Collaborating with Industry: Strategies for an Undergraduate Software Engineering Program," in *Proceedings of the 3rd International Workshop on Summit on Software Engineering Education*, pp. 13-16, Shanghai, China, May 2006
10. Women's College Coalition (http://www.womenscolleges.org/colleges/bycollege.htm)