Infusing Software Security in Software Engineering

Dr. Sushil Acharya, Robert Morris University

Acharya joined Robert Morris University in Spring 2005 after serving 15 years in the Software Industry. His teaching involvement and research interest are in the area of Software Engineering education, Software Verification & Validation, Data Mining, Neural Networks, and Enterprise Resource Planning. He also has interest in Learning Objectives based Education Material Design and Development. Acharya is a co-author of "Discrete Mathematics Applications for Information Systems Professionals- 2nd Ed., Prentice Hall". He is a member of Nepal Engineering Association and is also a member of ASEE, and ACM. Acharya was the Principal Investigator of the 2007 HP grant for Higher Education at RMU. In 2013 Acharya received a National Science Foundation (NSF) Grant for developing course materials through an industry-academia partnership in the area of Software Verification and Validation. Acharya is also the Director of Research and Grants at RMU.

Dr. Walter W Schilling Jr., Milwaukee School of Engineering

Walter Schilling is an Associate Professor in the Software Engineering program at the Milwaukee School of Engineering in Milwaukee, Wisconsin. He received his B.S.E.E. from Ohio Northern University and M.S. and Ph.D. from the University of Toledo. He worked for Ford Motor Company and Visteon as an Embedded Software Engineer for several years prior to returning for doctoral work. He has spent time at NASA Glenn Research Center in Cleveland, Ohio, and consulted for multiple embedded systems companies in the Midwest. In addition to one U.S. patent, Schilling has numerous publications in refereed international conferences and other journals. He received the Ohio Space Grant Consortium Doctoral Fellowship and has received awards from the IEEE Southeastern Michigan and IEEE Toledo Sections. He is a member of IEEE, IEEE Computer Society and ASEE. At MSOE, he coordinates courses in software quality assurance, software verification, software engineering practices, real time systems, secure software development, network security, and operating systems.

Infusing Software Security in Software Engineering

Abstract

Software is now ubiquitous and software security is now realized as a growing threat. It is important for software developers to fix software security problems, however more imperative is for software developers to understand that security features are not to be introduced as patchwork when a security situation arises but are to be addressed and handled very early in the software development lifecycle. Industry's general lack of ignorance of software security benefits and more importantly the shortage of software practitioners possessing software security understanding creates multitude of problems in the software industry. Imparting real world experiences in the academia as well as the industry is a challenge due to lack of effective active learning tools (ALT). Riding on the success of developing and disseminating, 42 delivery hours of active learning tools in the area of software verification and validation the authors propose to partner with industry to develop 14 delivery hours of course modules developing ALTs in the form of class exercises, case studies, and case study videos and delivering them using a flipped classroom model.

Through a gap analysis exercise jointly carried out with industry partners a draft requirements list has being identified. Specific exercises are being developed using an iterative development methodology. Student understanding is proposed to be assessed through quizzes, exams, assignment, and a learning survey. Once developed the ALTs will be made publicly available through a website. This paper discusses continuing work on the gap analysis in software security education, presents proposed contents areas for ALT, shares structures of three developed/proposed ALTs, presents a sample of a survey instrument, and presents a sample ALT on case study video.

1. Introduction and Rationale

Software is now ubiquitous and software security is now realized as a growing threat. Execution of insecure codes causes software security problems that lead to undesired consequences. Insecure Software Requirements Specifications (SRS) leads to insecure Software Design which eventually leads to insecure codes. Currently the industry focus is on fixing security flaws as they appear. In fact flaws arising from insecure code in released software products are now the highest unnecessary cost in software development. A report by Bank of America Merrill Lynch reckons that the cybersecurity market was $75 billion a year in 2015 and will be $170 billion by 2020. [1] It is important for software developers to fix software security problems, however more imperative is for software developers to understand that security features are not to be introduced as patchwork after a security situation arises but are to be addressed and handled very early in the software development lifecycle (SDLC).

In September 2014 a malware was inserted into Home Depot's computer network resulting in the siphoning of payment-card data and email addresses of 56 million customers. In 2015 the home-improvement chain had already spent $232 million as a result of this hacking [2]. In 2013 hackers broke into Target's network and accessed credit card information and other customer data of 70 million customers. The company said that in 2015 it booked $162 million in expenses across 2013 and 2014 related to this data breach [3]. Nortel, a Canadian telecoms giant, went bust in part because hackers stole its intellectual property and TalkTalk, one of the biggest phone and internet companies in Britain, is floundering after an attack which leaked customer information - which was apparently stored unencrypted, on a computer accessible through a public website. [1] This clearly shows a flaw in the design of the system and more important the ignorance of the practitioners. In the past five years these are simply examples of instances of private information being public because software developers were not able to do what the public had entrusted them to do. As we move into the era of Internet of Things (IoT) and edge computing the importance of developing secure software is even more pressing.

According to the 2004 *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* [4] Graduates of an undergraduate SE program must be able to meet 7 outcomes. Outcome 4 states "Design appropriate solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal, and economic concerns". It is through this outcome we expect students to design solutions that address ethical, social, legal, security, and economic concerns. The importance of security in the curriculum guidelines can be noted from the change in SE education Knowledge Areas. In the 2004 *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* [4] security was listed as an area of study. However in the 2014 *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* [5] there is now an increase in the visibility of software requirements and security. *Security is now one of the* 10 knowledge areas that make up the SEEK: computing essentials (CMP), mathematical and engineering fundamentals (FND), professional practice (PRF), software modeling and analysis (MAA), requirements analysis and specification (REQ), software design (DES), software verification & validation (VAV), software process (PRO), software quality (QUA), and security (SEC). The question now is are software engineering courses housed in Software Engineering, Computer Science or Information Systems programs reflective of this increased interest and concern on security.

In their paper "Software Engineering Education Needs More Engineering" [6] the authors point out that the 2004 *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* [4] for Undergraduate Degree Programs in Software Engineering appears not to be widely known or actively used. If that is the trend then the 2014 *Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* [5] may also be not widely known or actively used. If that is the case the significance of software security is questionable.

The fundamental challenge towards a solution that will improve software security lies in the people and processes that develop and produce software. Industry's general lack of ignorance of software security benefits and more importantly the shortage of software practitioners possessing software security understanding creates multitude of problems in the software industry. Imparting real world experiences in the academia as well as the industry is a challenge due to lack of effective active learning tools (ALT). At the author's institution, this educational resource gap is being addressed by developing ALTs in the form of class exercises, case studies, and case study videos and delivering them using a flipped classroom model. Riding on the success of developing and disseminating, 44 delivery hours of active learning tools in the area of software verification and validation the authors propose to partner with industry to develop 14 delivery hours of course modules in the form of active learning tools that can be incorporated in existing software degree courses. 6 delivery hours of case studies, 6 delivery hours of exercises, and 2 delivery hours of case study videos are being designed and developed. Through a gap analysis exercise jointly carried out with industry partners a requirements list is being identified. This process is on-going. Specific exercises are then being developed using an iterative development methodology depicting industry-academic partnership. Student understanding is proposed to be assessed through quizzes, exams, assignment, and a learning survey. Once developed the ALT's will be made publicly available.

This paper discusses developed and continuing work on software security ALTs. In section 2 gap analysis in software security education from industry and academia perspectives is presented. In section 3 proposed contents areas for the ALTs is discussed with emphasis on security touchpoints. In Section 4 three ALTs are described in detail. In section 5 ALT delivery strategy is briefly discussed followed by section 6 where students learning assessment is described. Finally in section 7 we present a sample case study video ALT.

2. The Gap in Software Security Education

2.1. Industry Survey

A detail survey is planned for the summer of 2017 however a preliminary survey consisting of 12 mostly "Yes" and "No" questions on the state software security knowledge in the context of new hires was conducted with 4 companies. The results of the survey are depicted in Table 1. In addition the response to two other questions pertaining to industry expectations from new hires in the software security field is depicted in Table 2.

From Table 1 and Table 2 we see that the industry expects new hires to have knowledge on software security. Some have been very specific on what they want in new hires and have specified security touch points (described later). Only one company said it found the person they needed. Two of the companies said there is demand for software individuals with security knowledge and they had a hard time finding the right person; one is still searching. From the tables it is observed that there is a gap between what the industry expects to see in new hires and what the students learn and retain coming out of school. The development and dissemination of ALTs and the incorporation of these in applicable software courses is expected to fill this void.

Table 1: Industry Survey Response (4 responses)

Questions	Yes	No
1. Are new hires aware of common types of security vulnerabilities? Yes or No?	20%	80%
2. Are new hires able to document security requirements? Yes or No?	20%	80%
3. Are new hires able to apply secure design principles to the creation of secure software? Yes or No?	20%	80%
4. Are new hires able to verify the security mechanisms implemented in their system? Yes or No?	20%	80%
5. Are new hires able to use misuse cases to document cases system should not allow? Yes or No?	20%	80%
6. In your view should new hires have knowledge of specific processes? Yes or No?	20%	80%
7. In your view should new hires have knowledge of specific tools? Yes or No?	100%	0%
8. Does your company set aside budget for training/mentoring for new hires that do not have basic knowledge on secure software development? Yes or No?	100%	0%
9. Do you think your company will benefit if new hires were taught the basics of secure software development as part of their undergraduate curriculum? Yes or No?	100%	0%
10. Does your company have a separate security organization? Yes or No?	100%	0%

Table 2: Industry Survey Response (4 responses)

11. In terms of Software Security what do you expect new hires to have knowledge of when they join your company? Processes? Methods? Tools?
• It is important for new hires to have general understanding of what secure coding is and how to prevent standard vulnerabilities. • New hires also need to have knowledge of processes, methods, and tools. Their understanding of industry standard best practices for security in the Software Development Life Cycle is critical. The importance of it as it relates to business risk, common use cases (Web Application, Software Applications, etc.) and exposure to the common tools or tool types and where they should be applied (e.g. DAST, SAST, IAST). • New hires need to understand technical capabilities to so that they understand why we care and how we prevent the vulnerabilities/threat associated with it. • It is surprising that even seasoned software and application developers don’t understand the basics of vulnerabilities, false positives and how to clean their code. This is a serious problem.
12. What security touchpoints does your company use to ensure security in developed products?
• Proper project management and security milestones in the SDLC is very important. Best practices and industry standards are available however most developers don’t know they should have the proper tools to scan their own code for vulnerabilities and make repairs, what proper configuration security means, etc. Most think security is just using proper passwords and that’s all. • We use Microsoft Secure Development Lifecycle. • We have security reviews of designs, architecture and coding. We also incorporate security testing of our applications. • Technical design review, Code review, All codes now go through static code analysis prior to production, Ethical Hacking team analyze and scan for vulnerability in lower environment prior to production

As the format of this survey did not enable participants to answer a “weak yes” for the detail survey planned for the summer of 2017 the questionnaire will be redesigned to make this option available. In addition questions asked to the industry will be mapped to the questions asked to the academia.

2.2. Academia Survey

A detail survey is planned for the summer of 2017 however a preliminary survey consisting of seven questions was conducted with 11 academic institutions on teaching software security in the academia. These institutions offer a degree or track in Software Engineering (SE), Computer Science (CS) and Information Systems (IS). The questions focused on teaching requirements and the lack of teaching modules. The results of the survey are depicted in Table 3.

Table 3: Academia Survey Response (11 responses)

Questions	Yes	No
1. Do you think it is important to impart knowledge on so Yes or No?	82%	18%
2. Is Software Security discussed in any core course at your university? Yes or No?	55%	45%
3. Do you think it is important to impart knowledge on Software Security to undergraduate students pursuing a degree in the Software field? Yes or No?	100%	0%
4. Is Software Security discussed in the software courses you teach? Yes or No?	46%	54%
5. Is it easy for you to find course materials or course modules related to software security? Yes or No?	0%	100%
6. Should course materials or course modules be engaging to students? Yes or No?	100%	0%
7. If course modules are available in the form of case studies, class exercises and videos would you incorporate one or more of these in your software course? Yes or No?	100%	0%

All faculty members from the institutions who participated in this survey say they are able to incorporate engaging materials in their current courses. All of them also agree that students pursuing a degree in a software field should have knowledge on software security. Not all of them agree that every major needs to have knowledge of software security. The authors however feel that imparting knowledge on security to all majors is necessary but feel the knowledge level should differ in comparison to those pursuing CS, SE and IS degrees. The survey also indicates that there is a need for engaging course modules. The development and dissemination of ALTs is expected to fill this void.

As the format of this survey did not enable participants to answer a “weak yes” for the detail survey planned for the summer of 2017 the questionnaire will be redesigned to make this option available. In addition questions asked to the mapped will be mapped to the questions asked to the industry.

3. Contents Coverage

Software security is not security software. According to McGraw [7] Software security rests on three pillars: applied risk management, software security touchpoints, and knowledge. Applied risk management deals with identifying, ranking, tracking, and understanding software security risks. Software security touchpoints deals with 7 software security best practices (see figure 1). And knowledge deals with gathering, encapsulating, and sharing security knowledge. By applying the three pillars in a gradual, evolutionary manner and in equal measure we can realize a reasonable cost effective software security program.

Software security is a vast topic that involves many processes, methods, and tools. Not all aspects can be taught in an undergraduate program. However the academia should be able to teach the basics so that the industry can easily pick up from where the academia left off resulting in graduates/professionals developing software with a security focus.

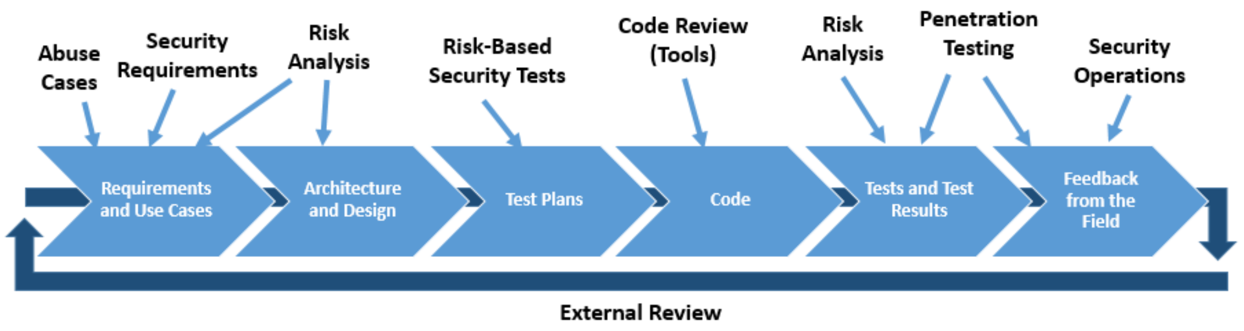


Figure 1: Security Touchpoints [7]

The authors are using the 7 security touchpoint depicted in Figure 1 to identify software security focus topic areas for our ALTs. They have also spoken to two companies, Eaton Corporation and ANSYS, and both have concurred with the authors and have expressed interest in partnering in identifying, developing, and verifying appropriate ALTs. Based on research and the authors conversation with Eaton Corporation and ANSYS the authors are confident that the ALTs will be on a subset of the following software security focus topics:

- Confidentiality, Integrity, and Availability
- Software Security Touchpoints
- Risk Management Frameworks
- Security Goals versus Security Functions
- Abuse and Misuse Cases
- Design Principles for security
- Threat Modeling
- Architectural Risk Analysis
- Identity Management
- Static Analysis of source code
- Defensive coding
- Security Testing
  - Penetration testing
  - Fuzz testing

4. Active Learning Tools

Active learning is "embodied in a learning environment where the teachers and students are actively engaged with the content through discussions, problem-solving, critical thinking, debate or a host of other activities that promote interaction among learners, instructors and the material" [8]. Specifically, active learning helps students develop problem-solving, critical-reasoning, and analytical skills, all of which are valuable tools that prepare students to make better decisions and become better students and, ultimately, better employees [9]. Active learning is achievable by complementing lecture materials with case studies, class exercises, and case study videos. The templates for the ALTs have been developed and are described below. The authors have already developed one ALT, a case study video, which is currently incorporated in a software engineering course at one of the author's institution.

4.1. Case Studies

Case studies can serve as useful tools to teach applications of science and engineering principles. In a study at Middlesex Community College [10], case studies were used in teaching General Biology - I where 88.2% of the students surveyed found the cases to be useful or better for learning the course content. 90.9% of the students surveyed thought the cases were useful or better in making the course more interesting. Case studies were applied in six courses to help students (1) understand complex and complicated issues and describe interrelated processes; (2) discuss policy- and decision-making ideologies that either are politically or socially charged; and (3) engage in informative and focused classroom discussion. The results indicated that use of the case study method as an active learning tool provides students with a variety of important skills necessary for success both in and out of the classroom. Acharya et. al. have successfully disseminated sixteen case studies in the area of software V&V [11]. The authors are developing 6 delivery hours of case studies with each case study module being 25 delivery minutes or multiple of 25 delivery minutes. Once completed each case study will consist of the following components:

a. **Case Study Description:** This document provides complete information of this active learning tool. It has four categories of information. The first part provides general information about the case study and includes details like the software security focus topic area, module name, prerequisite knowledge, learning outcomes, keywords, expected delivery duration, description of the scenes, and student exercise. The second part describes the instruction and assessment procedure. The third part has a list of possible discussion questions by scene. The final part of this document depicts the survey instrument.
b. **Student Handout:** Student Handout includes everything that the students need to participate in the class discussion. This handout explains the scenes, the objectives of the exercise, step-by-step what the student should do and a set of sample question for Scene 1.
c. **Discussion Questions:** For each video, suggested discussion questions for each scene is available as a power point slides. Instructors are welcome to modify these questions or use their own questions.
d. **Assessment Instrument:** The assessment instrument is a simple survey primarily for indirect assessment of student learning outcome, and also for student feedback. This is a survey that assesses students on communication and content knowledge. It is designed for generic use in every exercise, to be completed quickly at the conclusion of the class exercise.

4.2. Class Exercises

Class exercises provide class time to explicitly raise questions that invite student participation. When well designed for the context and presented in the right setting, class exercises raise questions for the students to exercise their thinking. Woods and Howard [12] effectively used class exercises for Information Technology students to study ethical issues. Day and Foley [13] used class time exclusively for exercises, having their students prepare for class with materials provided online. Frydenberg [14] primarily used hands-on exercises to foster student understanding in data analytics. Well designed, class exercises become very effective learning tools and can be versatile in various classroom settings. Acharya et. al. have successfully disseminated sixteen exercises in the area of software V&V [15]. The authors are developing 6 delivery hours of class exercises with each class exercise module being 25 delivery minutes or multiple of 25 delivery minutes. Once completed each exercise will consist of the following components:

a. **Class Exercises Description:** This document provides complete information of this active learning tool. It has four categories of information. The first part provides general information about the class exercise and includes details like the software security focus topic area, module name, prerequisite knowledge, learning outcomes, keywords, expected delivery duration, description of the scenes, and student exercise. The second part describes the instruction and assessment procedure. The third part has a list of possible discussion questions by scene. The final part of this document depicts the survey instrument.
b. **Student Handout:** Student Handout includes everything that the students need to participate in the class discussion. This handout explains the scenes, the objectives of the exercise, step-by-step what the student should do and a set of sample question for Scene 1.
c. **Discussion Questions:** For each video, suggested discussion questions for each scene is available as a power point slides. Instructors are welcome to modify these questions or use their own questions.
d. **Assessment Instrument:** The assessment instrument is a simple survey primarily for indirect assessment of student learning outcome, and also for student feedback. This is a survey that assesses students on communication and content knowledge. It is designed for generic use in every exercise, to be completed quickly at the conclusion of the class exercise.

4.3. Case Study Videos

One commonly used technique to enhance the classroom learning experience is the use of video. Videos are viewed as an effective method of presenting standard material while addressing students of different learning styles. A video engages visual learners with its images and motions, while auditory learners can listen carefully to the narration to gain an understanding of the topic. Videos are an essential part of the flipped classroom model, in which the preponderance of lecture material is presented before class [16]. Watching videos can reinforce reading and lecture material, help to develop common knowledge, enhance the quality of discussion and overall student comprehension, accommodate students of different learning styles, increase student motivation, and increase teacher effectiveness [17]. Videos can aid in showcasing highly complex concepts and ideas in a short period of time, provoking meaningful discussion and analysis. Acharya et. al. have successfully disseminated 4 case study videos in the area of software V&V [18]. One of these videos titled “Security Inspection Scenes” is currently being used as a software security ALT. The authors are proposing producing 1 more case study video. Combined both videos will be suitable for 2 plus delivery hours. Each case study video consists of the following components. The description below is based on the “Security Inspection Scenes” video. The authors expect the second video to follow the same format.

a. **Video:** Videos have appropriate narrations and pause points between scenes for incorporating class discussions.

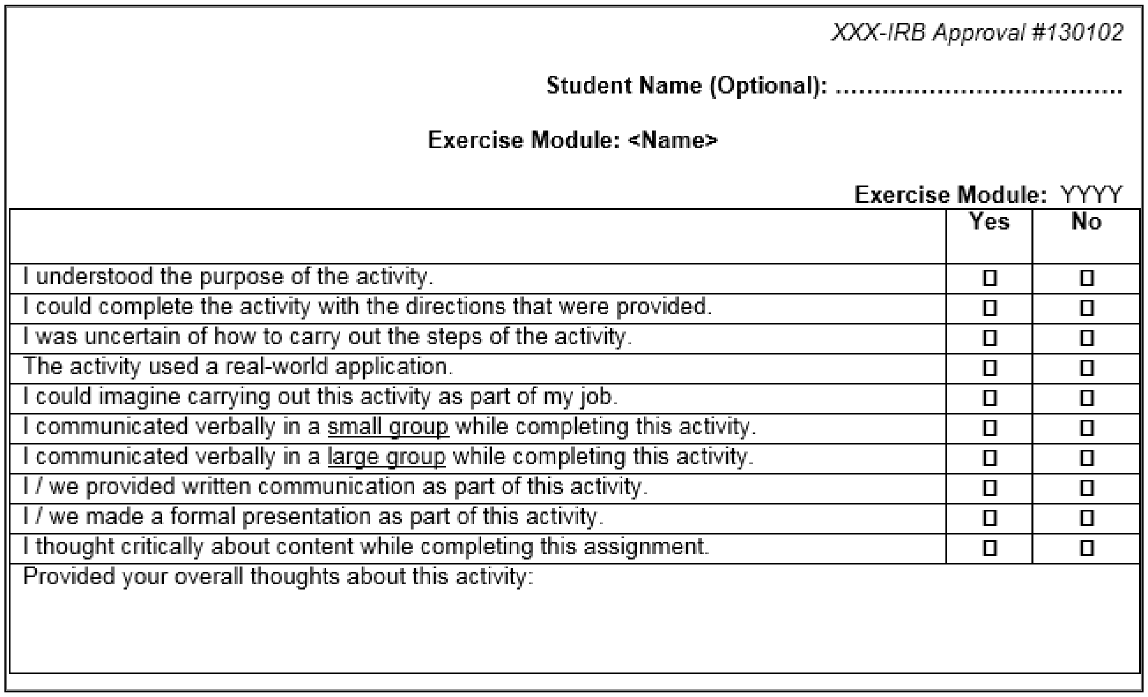
b. **Case Study Video Description:** This document provides complete information of this active learning tool. It has four categories of information. The first part provides general information about the video and includes details like the software security focus topic area, module name, prerequisite knowledge, learning outcomes, keywords, expected delivery duration, description of the scenes, and student exercise. The second part describes the instruction and assessment procedure. The third part has a list of possible discussion questions by scene. The final part of this document depicts the survey instrument.
c. **Discussion Questions:** For each video, suggested discussion questions for each scene is available as a power point slides. Instructors are welcome to modify these questions or use their own questions.
d. **Student Handout:** Student Handout includes everything that the students need to participate in the class discussion. This handout explains the scenes, the objectives of the exercise, step-by-step what the student should do, and a set of sample question for the first scene of the video.
e. **Assessment Instrument:** The assessment instrument is a simple survey primarily for indirect assessment of student learning outcome, and also for student feedback. This is a survey that assesses students on communication and content knowledge. It is designed for generic use in every exercise, to be completed quickly at the conclusion of the class exercise.

5. Delivery Strategy

With the active learning tools designed to impart practical knowledge into theoretical understanding, the authors encourage a flipped classroom model [19] in which class time for active learning tools can be maximized so as to engage the students for further digestion of the knowledge in the context of industry practices. Students are expected to be prepared outside of the classroom beforehand, with assigned textbook readings or reviewing of online materials. The class time is then spent on discussion and teamwork, reinforcing the material from the previous session. For effective delivery it is also recommended that students work in small teams. Overall, the flipped classroom model has proven highly effective at increasing student engagement and enhancing the preparation of students for class sessions [20]. The flipped classroom also has been shown to allow the instructor to cover more material and results in higher student performance [21].

6. Assessing student learning

While exams, quizzes will be used to understand student learning the survey is depicted in Figure 2 below will be used to understand student perception of the delivered active learning tools delivered. A survey will be conducted after delivery of each ALT.



XXX-IRB Approval #130102

Student Name (Optional): ………………………………

Exercise Module: <Name>

Exercise Module: YYYY

	Yes	**No**
I understood the purpose of the activity.	☐	☐
I could complete the activity with the directions that were provided.	☐	☐
I was uncertain of how to carry out the steps of the activity.	☐	☐
The activity used a real-world application.	☐	☐
I could imagine carrying out this activity as part of my job.	☐	☐
I communicated verbally in a small group while completing this activity.	☐	☐
I communicated verbally in a large group while completing this activity.	☐	☐
I / we provided written communication as part of this activity.	☐	☐
I / we made a formal presentation as part of this activity.	☐	☐
I thought critically about content while completing this assignment.	☐	☐
Provided your overall thoughts about this activity:		

Figure 2: Student survey

7. Sample ALT: Security Inspection Scenes

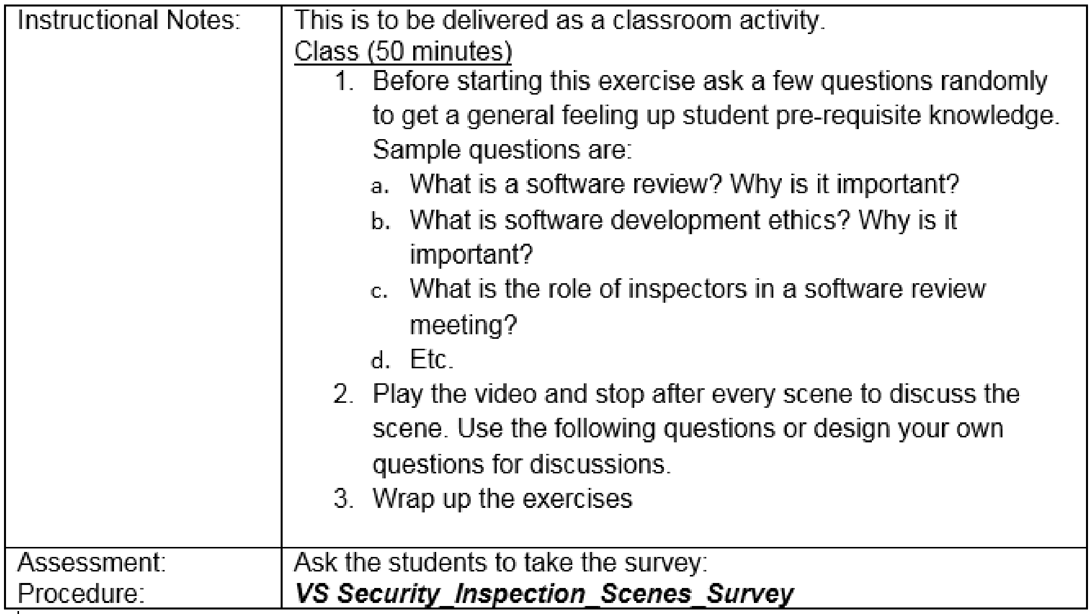
While this is paper is work in progress the authors have already started working on a few ALTs. Below we present sample components of an ALT on Security Inspection Scenes.

Introduction: The scenes in this case study video portray brief dramatizations in a Security Inspection case study. The scenes present industry best practices and problems that can occur during the process. The objectives of this ALT are:

1. Explain the potential impacts of a security violation on a deployed system.
2. Explain the impacts of improper risk assessment on security.
3. Explain the importance of a system architecture in the design of a software system.
4. Understand the importance of using proper protocols to secure a system.
5. Understand how threat modeling can be applied.

The components of this ALT are depicted below.

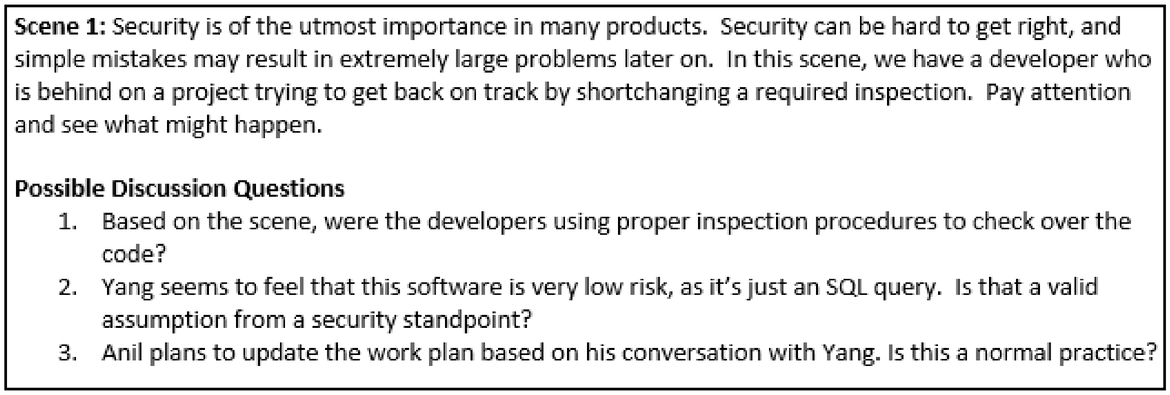
- **Component 1: Description Document**
  This document provides detail information of this ALT. Figure 3 illustrates the "Instructional notes" of this document.



Instructional Notes:	This is to be delivered as a classroom activity. Class (50 minutes) 1. Before starting this exercise ask a few questions randomly to get a general feeling up student pre-requisite knowledge. Sample questions are: a. What is a software review? Why is it important? b. What is software development ethics? Why is it important? c. What is the role of inspectors in a software review meeting? d. Etc. 2. Play the video and stop after every scene to discuss the scene. Use the following questions or design your own questions for discussions. 3. Wrap up the exercises
Assessment: Procedure:	Ask the students to take the survey: ***VS Security_Inspection_Scenes_Survey***

Figure 3: Instruction Notes

- **Component 2: Discussion Questions**
  Figure 4 depicts a scene description and discussion questions for this case study video.



Scene 1: Security is of the utmost importance in many products. Security can be hard to get right, and simple mistakes may result in extremely large problems later on. In this scene, we have a developer who is behind on a project trying to get back on track by shortchanging a required inspection. Pay attention and see what might happen.

Possible Discussion Questions

1. Based on the scene, were the developers using proper inspection procedures to check over the code?
2. Yang seems to feel that this software is very low risk, as it's just an SQL query. Is that a valid assumption from a security standpoint?
3. Anil plans to update the work plan based on his conversation with Yang. Is this a normal practice?

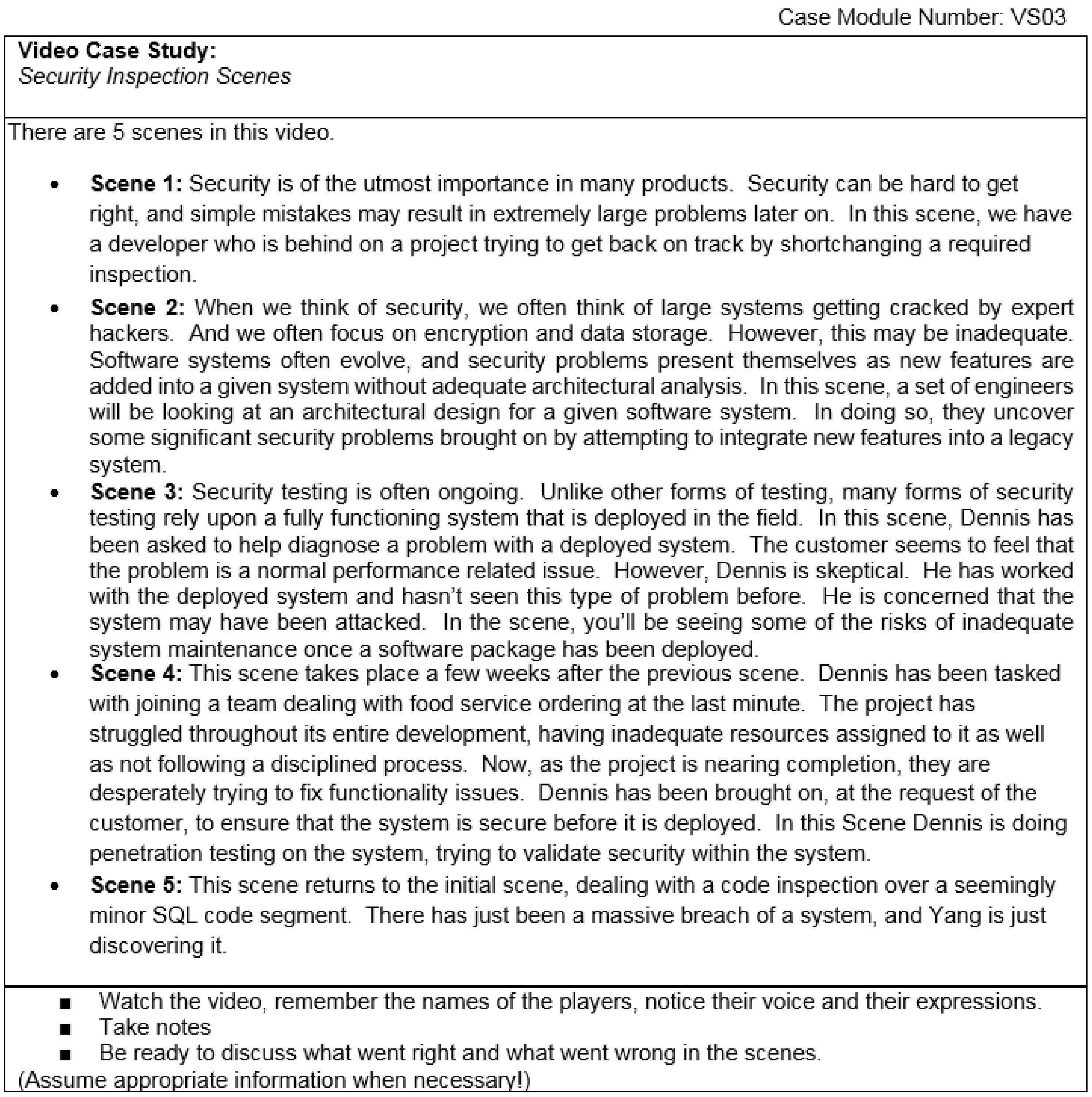
Figure 4: Scene and Discussion Questions

- **Component 3: Student Handout**
  Figure 5 illustrates a portion of the student handout.

- **Component 4: Assessment Instrument**
  Figure 6 illustrates the assessment instrument for this ALT.

8. Conclusion and Recommendation

Overall, the need for security education within the software engineering real is paramount. Having good, relevant materials is also essential. In developing relevant materials, it is important that the academic community solicit input from industry stakeholders. Industry stakeholders have a unique perspective into the needs of graduating students as well as the risks faced by deployed systems. To accomplish this, a set of industry partners has been surveyed and their responses categorized, focusing the development effort into the relevant areas. The team has also developed a strategy to use active learning tools, including case studies, class exercises, and case study videos, to help students understand the aspects of security. Preliminary assessment plans and general topical organizations have also been developed and a sample case study video is already incorporated in a software engineering course.



Case Module Number: VS03

Video Case Study:
Security Inspection Scenes

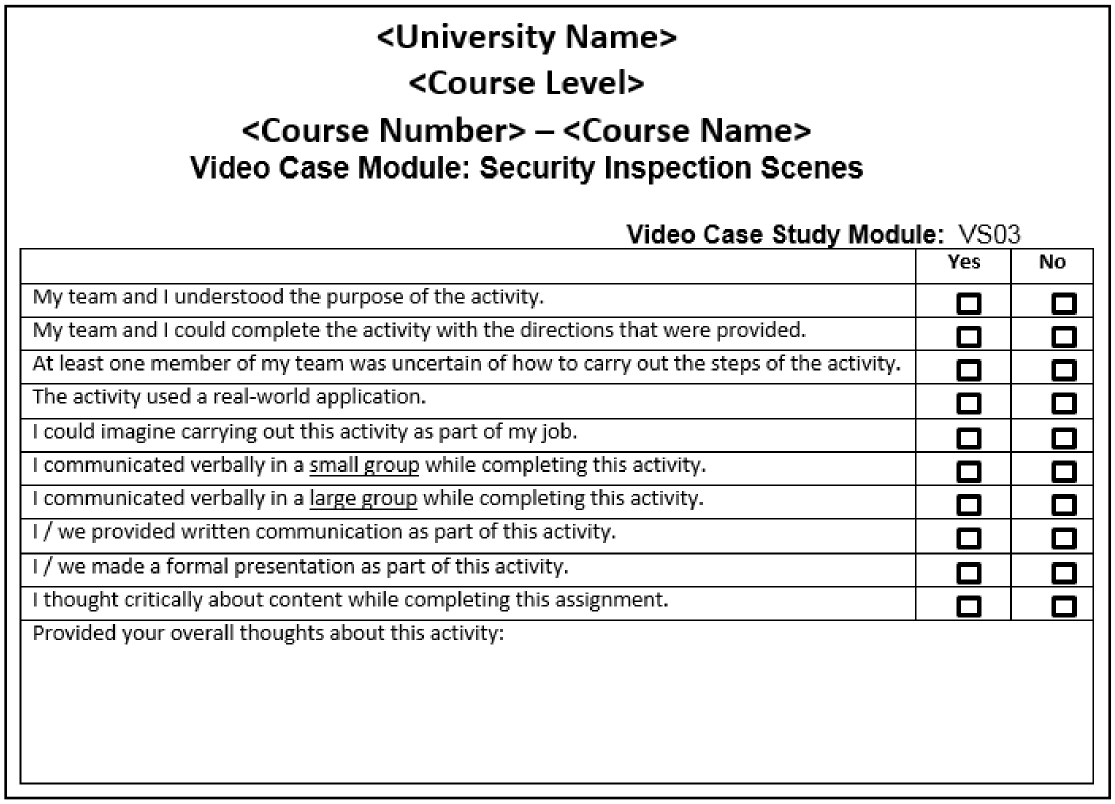
There are 5 scenes in this video.

- **Scene 1:** Security is of the utmost importance in many products. Security can be hard to get right, and simple mistakes may result in extremely large problems later on. In this scene, we have a developer who is behind on a project trying to get back on track by shortchanging a required inspection.
- **Scene 2:** When we think of security, we often think of large systems getting cracked by expert hackers. And we often focus on encryption and data storage. However, this may be inadequate. Software systems often evolve, and security problems present themselves as new features are added into a given system without adequate architectural analysis. In this scene, a set of engineers will be looking at an architectural design for a given software system. In doing so, they uncover some significant security problems brought on by attempting to integrate new features into a legacy system.
- **Scene 3:** Security testing is often ongoing. Unlike other forms of testing, many forms of security testing rely upon a fully functioning system that is deployed in the field. In this scene, Dennis has been asked to help diagnose a problem with a deployed system. The customer seems to feel that the problem is a normal performance related issue. However, Dennis is skeptical. He has worked with the deployed system and hasn't seen this type of problem before. He is concerned that the system may have been attacked. In the scene, you'll be seeing some of the risks of inadequate system maintenance once a software package has been deployed.
- **Scene 4:** This scene takes place a few weeks after the previous scene. Dennis has been tasked with joining a team dealing with food service ordering at the last minute. The project has struggled throughout its entire development, having inadequate resources assigned to it as well as not following a disciplined process. Now, as the project is nearing completion, they are desperately trying to fix functionality issues. Dennis has been brought on, at the request of the customer, to ensure that the system is secure before it is deployed. In this Scene Dennis is doing penetration testing on the system, trying to validate security within the system.
- **Scene 5:** This scene returns to the initial scene, dealing with a code inspection over a seemingly minor SQL code segment. There has just been a massive breach of a system, and Yang is just discovering it.

- ■ Watch the video, remember the names of the players, notice their voice and their expressions.
- ■ Take notes
- ■ Be ready to discuss what went right and what went wrong in the scenes.

(Assume appropriate information when necessary!)

Figure 5: Student Handout



<University Name>

<Course Level>

<Course Number> – <Course Name>

Video Case Module: Security Inspection Scenes

Video Case Study Module: VS03

	Yes	No
My team and I understood the purpose of the activity.	☐	☐
My team and I could complete the activity with the directions that were provided.	☐	☐
At least one member of my team was uncertain of how to carry out the steps of the activity.	☐	☐
The activity used a real-world application.	☐	☐
I could imagine carrying out this activity as part of my job.	☐	☐
I communicated verbally in a small group while completing this activity.	☐	☐
I communicated verbally in a large group while completing this activity.	☐	☐
I / we provided written communication as part of this activity.	☐	☐
I / we made a formal presentation as part of this activity.	☐	☐
I thought critically about content while completing this assignment.	☐	☐
Provided your overall thoughts about this activity:		

Figure 6: Case study video assessment instrument

References

[1]. The Cost of Immaturity, [Web Page], Retrieved January 11, 2017 from URL http://www.economist.com/news/business/21677639-business-protecting-against-computer-hacking-booming-cost-immaturity

[2]. Home Depot Breach Cost Expected to Reach Billions, [Web Page], Retrieved January 11, 2017 from URL https://www.scmagazine.com/home-depot-breach-costs-expected-to-reach-billions/article/533722/

[3]. Target Says Credit Card Data Breach Cost it $162 Million in 2013-14, [Web Page], Retrieved January 11, 2017 from URL https://techcrunch.com/2015/02/25/target-says-credit-card-data-breach-cost-it-162m-in-2013-14/

[4]. The Joint Task Force on Computing Curricula: IEEE-CS and ACM, (2004), *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*, [Webpage], Retrieved January 29, 2017 from URL http://sites.computer.org/ccse/SE2004Volume.pdf

[5]. The Joint Task Force on Computing Curricula: IEEE-CS and ACM, (2015), *Software Engineering 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*, [Webpage], Retrieved January 29, 2017 from URL https://www.acm.org/education/se2014.pdf

[6]. Acharya, S., Ackerman, A. (2012), Software Engineering Education Needs more Engineering, ASEE Annual Conference & Exposition – Software Engineering Constituent Committee, June 10 – 13 – San Antonio, TX

[7]. *McGraw, G. Software Security - Building Security In,* Addition-Wesley Software Security Series, ISBN: 0-321-35670-5, 2006
[8]. Promoting Active Learning, [Web Page], Retrieved January 21, 2017 from URL https://utah.instructure.com/courses/148446/pages/active-learning
[9]. Kunselman, J.C. and Johnson, K.A., Using the Case Method to Facilitate learning, College Teaching, Vol. 52. No. 3 (Summer 2004).
[10]. [Web Page], Retrieved December 2, 2016 from URL https://www.middlesex.mass.edu/sotl/downloads/klein.pdf
[11]. Manohar, P., Acharya S., Wu P.Y., Hansen M, Ansari A.A. & Schilling Jr W.W. (2015), Case Studies for Enhancing Student Engagement and Active Learning in Software V&V Education, Journal of Education and Learning, Vol 4, No. 4, December 2015, Page, 39-52
[12]. Woods, D., and Howard, E (2014) An Active Learning Activity for an IT Ethics Course. Information Systems Education Journal, 12(1) pp.73-77. http://isedj.org/2014-12/ ISSN: 1545-679X.
[13]. Day, J.A. and Foley, J.D. (2006) Evaluating a web lecture intervention in a human–computer interaction course. IEEE Transactions on Education, 49(4):420–431, 2006.
[14]. Frydenberg, M. (2013) Flipping Excel, Information Systems Education Journal, 11(1) pp.63-73. http://isedj.org/2013-11/ ISSN: 1545-679X.
[15]. Wu, P., Manohar, P. A., & Acharya, S. (2016). The Design and Evaluation of Class Exercises for Active Learning in Software Verification and Validation. *Information Systems and Education Journal*, *14*(4), 4-12.
[16]. Bergmann, J. and Aaron S. (2012) *Flip Your Classroom: Reach Every Student in Every Class Every Day*, Eugene: International Society for Technology in Education, 2012. Print.
[17]. Corporation for Public Broadcasting. (2004). Television goes to school: The impact of video on student learning in formal education
[18]. Acharya, S., Manohar, P. A., & Wu, P. (2016). *Using Case Study Videos to Effectively Teach Software Development Best Practices* (pp. 230-235). The 20th World Multi-Conference on Systemics, Cybernetics, and Informatics (WMSCI) Conference, Orlando, FL, Organized by International Institute of Informatics and Systemics (IIIS).
[19]. Bonwell, C. C., & Eison, J. A. (1991). *Active Learning; Creating Excitement in the Classroom*. ASHE-ERIC Higher Education Report No. 1., Washington, D.C.: School of Education and Human Development, The George Washington University. Retrieved from http://files.eric.ed.gov/fulltext/ED336049.pdf
[20]. Day, J.A. and Foley, J.D. (2006) Evaluating a web lecture intervention in a human–computer interaction course. IEEE Transactions on Education, 49(4):420–431, 2006.
[21]. Mason, G.S.; Shuman, T.R.; Cook, K.E. (2013), "Comparing the Effectiveness of an Inverted Classroom to a Traditional Classroom in an Upper-Division Engineering Course," *Education, IEEE Transactions on* , vol.56, no.4, pp.430,435, Nov. 2013