3815ICT-Software Engineering Workshop 1

Activity 1

Read the following two online articles.

1. www.toptal.com/swift/from-objective-c-to-swift

2. www.infoworld.com/article/2920333/mobile-development/swift-vs-objective-c-10-reasons-the-future-favors-swift.html

Write 2 paragraphs, about 15 lines in total, describing the features of Objective-C or/and Swift that lead to better quality software (less faults).

Response 1

Objective-C is the original programming language utilized to create applications for first generation iPhones under the iOS operating system. Since then, Apple has developed a more dedicated programming language which is tailored for iOS and Apple products. Swift is essentially a faster, safer and more succinct language which allows for less code being produced to accomplish the same task as its former, objective –C. Objective –C was based off of the language C. C is a very old programming language and as an early iteration of a programming language, suffers from oversights in its development simply due to the infancy of software development and frameworks at the time. Many of these shortcomings have been transferred to objective –C. Being a programming language developed from scratch, Swift has been moulded by modern day experts to fit specifications necessary for use with iOS and has trimmed the fat from older more clunky languages.

Objective –C was privy to bugs as a result of the handling of the 'null' pointers. This has been changed in Swift which in a very basic instance, provides a safer programming environment. Objective –C does not even hold the capacity to concatenate with a '+' sign. Swift, being a more modern language does support this. Simple features have been incorporated into swift which means that less code is required to accomplish tasks. Another example of this simplification is in the two-file system required to efficiently run an objective –C program. Essentially, 2 files must be utilized side-by-side where as swift simply requires one. There are various features of objective –C which might be considered archaic and swift has been developed to modernise programming.

Activity 2

Review the following tutorials and start using git.

If you already know and prefer to use cvs or subversion, that is fine. Just be aware that for your project you are required to use some Version Control System.

1. guides.github.com/introduction/git-handbook/
2. www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud

Write 3 benefits of using a version control system (no more than 200 words in total).

Response 2

A Version Control System (VCS) is utilized to collaborate on projects with multiple users. All documents are stored in a centralised location and can be accessed and edited by all collaborators. Each update to the system will be recorded and every update will be accessible. Furthermore, information will be recorded such as the user who issued the update along with a timestamp.

The benefits of a system such as this speak for themselves.

1. An entire timeline of changes is visible and accessible in the case of loss of data. Roll-backs to earlier instances of a project or document are simple.
2. Accountability for each person who makes a change to the document.
3. Collaborators can work asynchronously from locations around the world.

Activity 3

These projects have been cited as projects that have caused significant damage to the reputation of software development professionals and companies.

1. The Space Shuttle Software costed $10 Billion USD, (this is millions of dollars more than planned). It was delivered 3 years late. It had serious quality issues: The first launch of Columbia was canceled because of a synchronization problem with the Shuttle's 5 on-board computers. The error was traced back to a change made 2 years earlier when a programmer changed a delay factor in an interrupt handler from 50 to 80 milliseconds. The likelihood of the error was small enough, that the error caused no harm during thousands of hours of testing. Substantial errors still remained while software in operation: Astronauts were supplied with a book of known software problems named "Program Notes and Waivers".

2. The Canadian Government portal. Original plan was to consolidate 1,500 Canadian government websites into a single portal on a single platform. In over three years, only 10,000 webpages of a total 17 million had successfully been migrated. The original budget was $9.4 million and soon was revised to $28 million.

3. In 2013, the BBC has scrapped a 98m digital production system, which its director general said had "wasted a huge amount of license fee payers' money". The Digital Media Initiative was set up in 2008 but was halted in autumn 2013 having never become fully operational. "I have serious concerns about how we managed this project," BBC director general Tony Hall said. An independent review has been launched "to find out what went wrong and what lessons can be learned", he said. The Digital Media Initiative (DMI) was intended to transform the way staff developed, used and shared video and audio material and was seen as an important part of a move of resources to Salford.

Choose one case, then write 2 paragraphs (10 to 15 lines) to discuss what potential software engineering principles or guidelines reported in the media as being violated and leading to the resulting failures. Find in the WEB a description of the software crisis.

Response 3

The Canadian government initiative to consolidate all government web pages into a single domain was a high-profile IT failure. The budget was blown out by millions of dollars because the scale of the work necessary to migrate all of the required documents was severely underestimated. The problem began in the planning phases. Successful software engineering is heavily influenced by the correct planning, modelling and conceptual development prior to actual implementation and in the case of the Canadian Government migration; it is clear that this was not competently actioned.

Adobe failed to deliver the project within the agreed timeframe and failed to adhere to the projected budget. It can only be interpolated that the necessary due-diligence was not undertaken while projecting this budget and the work was not done to adequately document the difficulties and risks associated with such a large-scale migration of services to a single domain. Before implementation, engineers should have spent more time measuring these risks along with the requirements necessary to deliver a project within the budget and agreed time-frame.

Activity 4

Review the following.

1. The report of an interview with Jennifer Lynch: www.infoq.com/articles/standishchaos-2015 on the “Standish Group 2015 Chaos Report”.

2. The video “The art of doing twice as much in half the time” www.youtube.com/watch?v=s4thQcgLCqk by Jeff Sutherland (en.wikipedia.org/wiki/Jeff_Sutherland).

Find in the WEB a copy of the Agile Manifesto.

Write 2 paragraphs, (10-15 lines) about why software development projects still fail. That is, argue why software building is a complex activity.

Response 4

The definition of a successful project according to the Chaos report is that it must be:

1. Delivered on Time

2. Adherent to Budget
3. Fulfilling the Requirements

Software building involves a complex team of experts who must facilitate the transition of software from the initial concept through to the finished product. This involves conceptualisation, documentation, modelling, development of architecture, prototyping, building, testing, optimising and retesting. If at any stage the timeframe for any one of these activities is blown out, so too will the overall timeframe for the project, thus, the budget can quickly run out of control.

Communication must be maintained between different groups within the team (which may be hundreds of people) and also the client (delivering specifications). Successful delivery of the requirements of a project hinges on communication between the development team and that client. If the client does not articulate their wishes adequately or the developers are complacent, the project may be considered a failure due to the software not fulfilling the intended requirements.

A Program may contain millions of lines of code and it is difficult to predict what may or may not hinder the progress of a project along with the magnitude of this hindrance. As such, Software Development Projects are still prone to failure.

Activity 5

Provide a simple justification.

1. In 1988, the Morris worm exploited a buffer overflow flaw in fingerd. It also exploited flaw in sendmail. The set up for sendmail enabled it with root privilege when run in debug mode. The worm spread by guessing user passwords. It is claimed the worm exploited a detachment from (violation) the following principles: “Principle of least privilege” and “Separation of privilege”, why?

2. In UNIX, when a process tries to read a file, the system checks access rights. If allowed, it gives the process a file descriptor. This file descriptor can be presented to the kernel for access. Note that if permissions are subsequently disallowed, the process still has the valid file descriptor! It is claimed it detached from (violated) the following principles: “Principle of Complete Mediation”, why?

3. The Melissa virus/worm (1999) was a MS-WORD macro. When the file opened, it would create and send an infected document to names in the user’s Outlook Express mailbox. The recipient would be asked whether to disable macros(!). If macros were enabled, the virus would launch. It is claimed it exploited a detachment from (a violation of) the following principles: “Principle of Psychological acceptability ” and “Principle of least privilege”, why?

4. The Chernobyl virus (1998) infected programs, when the program ran, the virus becomes resident in memory of the machine. Rebooting did not help. The virus writes random garbage to the hard drive. The virus attempted to trash FLASH BIOS. It is claimed the virus

exploited detached from (violated) the following principles: “Principle of Least Privilege”, “Principle of Fail-safe defaults”, and “Principle of complete mediation”. Why?

Write one paragraph, about 7-10 lines, explaining what seems to be the security principle that was not considered. Write one paragraph for each case of justification.

Response 5

1. The worm was able to assume root privilege when the correct triggers were activated. This is a violation of the Principle of Least Privilege as access to this privilege should be restricted unless otherwise necessary. Root privilege is the highest order privilege and this should be restricted heavily, for a worm to assume this privilege it needed to subvert the software’s inbuilt security implementations.
2. Access to objects must be checked to ensure they are allowed, the UNIX process was granted a token which allowed it to bypass subsequent checks. This in itself is dangerous as verification should always be a priority in security. The second flaw which led to the violation was the fact that the token was not revoked upon a change of permissions. This is a blatant oversight and easily exploitable.
3. The Melissa virus received privilege which allowed it to propagate itself. Aside from this, the virus was activated upon opening of a document. The user was prompted and if disabling macros, the virus would be able to invade the current user’s computer. It would then receive the necessary privilege to write and send another infected document, this level of privilege should only be granted to the user with the correct permission but this was not the case.
4. The Chernobyl virus was granted write privilege to the hard drive and bombarded the drive with useless data. This privilege should not be granted. According to the principle of fail-safe defaults, unless explicitly granted access, a subject should be denied. This was not the case with the virus. Finally, the virus should have been continuously checked throughout its lifecycle in order to maintain its privilege.

Activity 6

Write 15 lines of a reflective report on the previous activities. Analyse and evaluate the match of the activities to the learning objectives proposed in this workshop/laboratory.

Response 6

Each activity was essentially tailored to address a specific learning objective. The activities certainly covered a wide-range of topics from various languages and version control to security. I definitely feel it was helpful to have a general overview of topics which will be important to the Software Engineer’s skillset. It was also reasonably interesting to research various blunders in either security or development. I definitely feel as if the learning

objectives were adequately addressed in the activities presented. My preference for learning errs more toward the agile method of Software Development as I feel it is very innovative and I enjoyed the Ted talk delivered by its conceptualiser. I did find it extremely difficult to find a large amount of meaningful information about the Canadian IT implementation disaster and found that reading between the lines of the articles I found was necessary to ascertain how the Development process would have run. It is also convenient to be working with GitHub as 3 of my subjects this trimester will require the use of Version Control.