Jacob Steckel

Final Exam

Spring 2018

CSC402

Jacob Steckel

Article Review 11

CSC402

Applying the PSP in Industry

This article is about the Personal Software Process (PCP) and how it is impacting industries today. There are three main parts of the article that I feel are the most meaningful. First, why should an organization implement something like the PSP; what will it help with. Next, a real-life example of a company implementing the PSP is recorded. The reader gets to see first-hand what changes are made to certain procedures throughout the company. Lastly, the results of the PSP are given.   
 There were several reasons why the Personal Software Process was chosen as the next improvement technology. One of the goals were to give each software engineer a personal process that is reviewed and altered by the user. When it is partially defined by the engineers themselves, they can ensure that the process will suit their needs. The PSP is also used to provide the company with the opportunity to allow assessments to be reviewed individually rather than having company-level assessments. In the end, the ultimate goal of the PSP is to motivate software engineers to improve software quality.

Now for the actual implementation of the process. To train the engineers about the Personal Software Process, the company will require them to attend lectures and seven to ten exercises. The goal of the training is to introduce the new techniques and ideas and to begin forming good habits among the engineers. When it came to independent personal processes, the company modified the PSP to consider two different projects; features and fixes. They found that this was better than allowing everyone to modify their own lifecycle. When it came to fixes, they decided to separate them into two different categories: quick fixes and fixes that will take large effort and reengineering. They did this because each fix could vary depending on the time and difficulty of it. For quality management, the company kept what was suggested by the PSP and introduced personal code reviews. Personal code reviews will help improve the overall quality of the code at each stage of development. For managers, they were given two levels of measures, team-level and personal. For tool support and privacy, the organization made sure that the software engineers had sufficient tools so they could operate as efficiently as possible.

After the Personal Software Process was implemented and executed in the company, the results were recorded. Regarding the estimation and planning portion of the PSP, the process failed. The reason it failed was because it was very difficult making the participants work cohesively in a team environment. When it came to effort logging, engineers initially were logging approximately twenty actions a day. One year later, they were only logging ten actions a day. As time went on, they stopped logging altogether because they felt that it caused unneeded interruptions with their work. The one place that the PSP did work was quality management. The company already had some of the quality management ideas in place, but the PSP further improved what they already had. This article helped me learn a lot about how a company can introduce a software process and the effects that it could have.

Jacob Steckel

CSC402

Article Review 12

The Team Software Process: A Quiet Quality Revolution

The article begins by talking about how an exchange company named Electronic Broking Services (EBC) needed some software updates, but they did not want the updates to cause temporary delays in their services. To give a little background on EBC, their system handles between $80 billion and $100 billion worth of trades a day. The update was called Brokernet and it used the Team Software Process. Brokernet went so well while using the Team Software Process that EBC was immediately a believer and decided to use the process to update dozens of other programs.

The Team Software Process was initially developed by a man named Watts Humphrey. Humphrey was IBM’s director of programming quality and process. He used the Capability Maturity Model and improved it to make the TSP, which is now used by some of the top organizations in the world. Software development chief at Hill Air Force Base, Dave Webb, says that the TSP can be described quite simply. It’s a combination of good personal data and good project management techniques. The Hill Air Force Base used the TSP to create a program named TaskView, which automates the flight-planning process for the United States Air Force. Webb found that TaskView caused productivity to soar to 16% above the average compared to their other software process. Also, test time was reduced from 22% of the project schedule to only 2.6%.

Even though the Team Software Process has been shown to improve productivity and quality, there are a few reasons why it hasn’t taken over the development world completely. For one, the process is relatively new, so many organizations have yet to update and reevaluate its software process. Also, the prerequisite for the TSP is the PSP; the costs of training and implementing the PSP can cause companies to second guess whether they can afford this change. Smaller organizations many not have the time or funds to fully commit to an updated software process. The cost of the PSP training course is $4000 for the US government and academic organizations, $5000 for US industrial organizations, and $10000 for international organizations. On top of that, TSP coach training costs $3200 for the US government and academic organizations and costs $4000 for other US industries. So, although the positive effects are undeniable, the costs can be substantial. However, one expert claims that you can basically make your money back after 1,200 lines of code, so it is well worth it.

Many TSP supporters are shocked that more organizations aren’t taking advantage of this process. The increase in efficiency and quality are well worth whatever the potential costs are. If they want a competitive advantage over their competition, they better implement the TSP. At a company named Teradyne, software averaged about 20 test defects per KLOC. After they implemented the TSP, that number dropped to just 0.4 defects per KLOC. The effects of software processes such as the TSP are fascinating. It is incredible how updating your software process could make such a positive impact.

Jacob Steckel

CSC402

Article Review 13

Bridging the Gap

This article is about how large software development organizations traditionally distance themselves from information security and ignore the issue altogether. Many software development corporations seem to lack any type of understanding when it comes to technical security risks. This is leading to software being created that has tons of security weaknesses that could have been avoided if the company just put a little bit more of an emphasis on information security.

An increasing number of software developers are attempting to adopt software security procedures, but most lack the security domain knowledge to perform them sufficiently. For software to maximize its information security, companies must hire specialists who have years of knowledge dealing with system intrusions, malicious hackers, and software vulnerabilities. Every-day software developers typically do not have this type of knowledge and security experts are rarely involved in major software development projects.

The article goes into many of the security touchpoints that are available today. Abuse cases should be tested when software is being developed. This means that a specialist will deliberately try to misuse the program and will evaluate how the program responds. Using abuse cases can help figure out whether the software could possibly be used for malicious purposes.

At the design level, specialists must analyze business risk and architectural risk. For business risk, the specialist will consider how a project could possibly cause liability or loss of production for the parent organization that is sponsoring the project. Like business risk, architectural risk evaluates the security exposures of the software’s design and links them to business impact.

There are two types of procedures specialists must assess involving test planning: security functionality testing and risk-driven testing. Security functionality tests to make sure that the security features are secure by using methods such as encryption to protect user’s identification and privacy. With risk-driven testing the specialist acts as a “bad guy” and tries to exploit the software to find weaknesses and vulnerabilities.

The specialist should also implement a code review of the software where the code is evaluated to look for bugs and flaws in the software design.

For system testing, the specialist should perform system penetration testing. This type of testing focuses on human and procedural failures in the software’s configuration and deployment.

Lastly, the specialist should design a well-made deployment environment for a program that follows a software process. This could greatly enhance the programs overall security. Hopefully software development companies begin to take their information security seriously because it is a growing issue in the computer science world.

Jacob Steckel

CSC402

Article Review 14

SAGE: Whitebox Fuzzing for Security Testing

The article begins by talking about why Windows is constantly updating their security patches and the costs that they spend trying to protect users from malicious content. Hackers are always monitoring software products and looking for security vulnerabilities through two different methods. First, they inspect the code through binaries by using a disassembler. The other way is by blackbox fuzzing. Blackbox is a simple but effective way of finding vulnerabilities and it has been used to find 1000s of issues in software. It works by randomly mutating well-formed program inputs and then tests the program using those modified inputs, hoping to trigger a bug.

To combat the issue of blackbox fuzzing, Windows began developing an alternative to it called whitebox fuzzing. Whitebox fuzzing works by symbolically executing a program with well-formed input. The program is tested dynamically and it gathers constraints on inputs from branches that were encountered during execution. The gathered constraints are systemically negated and solved using a constraint solver. This process is performed continually using simple search techniques that try and go through all the execution paths of the program. While this is occurring, the search techniques will be checking several properties using a runtime checker. In theory, the systematic dynamic test generation will cover the entire program’s path. In reality, the search is usually not 100% complete because symbolic execution and constraint solving can be a bit inaccurate due to complex program statements. Also, the amount of executions paths in a program tend to be very large.

The first tool to utilize whitebox fuzzing was called the Scalable Automated Guided Execution, or SAGE. SAGE worked by negating all the constraints in a path one by one, and placing them in a conjunction with the prefix of the path constraint leading to it. Then, the constraint solver would attempt to solve it. This would lead to the single symbolic execution generating thousands of new tests.

The impact of SAGE has been undeniable. SAGE has found bugs in several different Microsoft applications. Many of these bugs were found in image processors, media players, file decoders, and document parsers. It has been recorded that SAGE is responsible for discovering approximately one-third of all the bugs discovered by file fuzzing since 2007.

SAGE has been huge for Microsoft. It is a combination of program analysis, testing, verification, and model checking techniques that have been established through the years. The way one tool can impact such a huge company is incredible to me, I found this article to be a very interesting read.

Jacob Steckel

CSC402

Article Review 15

Software Reviews: The State of the Practice

Software reviews have shown to be an effective practice when it comes to limiting software defects. Some observers estimate that review methods such as walkthroughs and software inspections can detect between 50 and 90 of a software’s defects. A software review is an assurance method that is a non-execution based approach used for identifying software defects, deviations from development standards, and many other problems. Although software reviews have been shown to be effective, many companies are still not buying into them. This article goes over a survey that was conducted in 2002 based around software reviews and their implementation by companies.

What is a software review? It is when a company examines its software projects and focuses on finding defects in the program. The examination is typically conducted by the developers, rather than an external group. The software review is a part of the development process, so development is not finished until a review is held.

For the survey, they invited 865 people from all over the world including many people from Germany and other parts of Europe. 226 people ended up responding and taking part in the survey. Companies of all different sizes participated, so there was a lot of variety. Some organizations were as small as 5 employees and some were as large as 10000 employees. 47% of the respondents claimed that they were of fewer than 500 employees.

The survey covered a variety of spectrums involving software reviews. When asked about what they were hoping to get out of software reviews, 74% said quality improvement and 54% said enforcing standards. Regarding the development process, 40% regularly review requirements and 30% regularly review code. Between 20-40% review the individual steps of their processes on a regular basis. For planning, only 20% typically perform planning and 30% don’t use any formal entry or exit criteria for their reviews. 40% of the companies have a preparation phase. For those who do, 50% use checklists and 35% use ad havoc reading. 40% of the surveyed companies hold a meeting during their software development. 35% provide some type of follow-up step after their software reviews. Thankfully 60% of the companies collect some sort of data, but 30% of those companies conduct no analysis and less than 25% utilize the data to perform reviews.

The survey showed two main conclusions. First, many of the companies that were surveyed included software reviews into their software development projects. Secondly, the review approaches tended to all be very much different from one another. There was no real systematic approach when these reviews were created. I am curious to see which companies had more of an in-depth approach to software reviews and how their company was impacted by it. The survey was intriguing, but I wish I could have seen the data itself so I could compare how different companies conducted their reviews.

Jacob Steckel

CSC402

Article Review 16

Industry Trends 2017

This article talks about many of the industry and market trends that are occurring in 2017. They ask questions such as what is most important for your company? And what do you feel is essential for your company to succeed?

The article begins by explaining where the industry is heading. The two biggest priorities for businesses and development teams right now is improvement development and IT efficiency. The products that an organization provides must meet increasing quality requirements, come with as little cost as possible, and have the ability to adapt to multiple different environments. New up-and-coming companies are entering the market with brand new ideas and solutions without any prior legacy in products. So, since change is indubitable, organizations that lack adaptability will stagnate and eventually fail.

The article goes into a survey that they conducted with their clients worldwide. They asked, what are the short-term and mid-term trends in your respective industry? There were a variety of answers, but a few tended to be much more common than others. If we were to sum up the obvious focus for companies, it’d be efficiency and innovation. The article states “a focus on efficiency and innovation is mandatory to survive in competitive markets”. That was the overall takeaway of the survey, but there were many smaller points that were also made evident.

To go into specifics, efficiency and cost was surveyed to be the clear number-one short-term goal for companies. Companies are willing to invest heavily into new innovative solutions if they are going to provide them with improved efficiency. They also place a large importance on cost reduction, so they hope that their investments eventually lead to lower costs. Regarding mid-term focuses, security and safety is the most important. Companies know that without proper security, their business would fail. Clients will not be willing to work with an organization if they know that their product has clear vulnerabilities.

A stable trend among all industries is innovation. With a focus on innovation, companies will continue to grow, strive to reduce costs, and improve their standards. Another trend that is growing is digital transformation and connectivity because this will impact the future business models of all industries. Governance and compliance is a growing trend because a growing number of standards will lead to more security and safety regulations. Another trend, distributed development, is now a prerequisite for nearly all companies. One trend that has been losing steam as of late has been complexity management because organizations have done a good job of implementing effective product management to control complexity. I enjoyed this article because I am interested in working for a startup or starting my own company at some point. So, learning about industry trends is something that will be beneficial to me.

Jacob Steckel

CSC402

Article Review 17

Looking into the Future

This article is centered around the future of software technology businesses. There are three main sections of the article that I will be covering because I feel like they are the most important. First, regarding the future of software, there are five dimensions that are given with examples included. Next, the article reveals the four major obstacles that a company must endure to become successful. Lastly, there are five vital factors that a company can use to find success through their future of challenges.

The five dimensions that characterize the future of software are collaboration, comprehension, connectivity, cloud, and convergence. A few examples for collaboration include: social networks, product and service, digital money, and crowdsourcing. Examples of comprehension include: data analytics, data quality, semantic search, and big data handling. For connectivity, some examples include: machine-to machine communication, mobile services, and sensor networks. A few examples of the cloud are: applications and services in the cloud, energy efficiency, sustainability, and location-based networks. For convergence, examples include: autonomous systems, the Internet of Things, and bioinformatics. These dimensions all call for new and improving software solutions.

There are four key obstacles that organizations will face in the future of software technologies. One obstacle could be a dysfunction organization that is unsure of their responsibilities. This could lead to a lack of focus and poor scheduling techniques. Another issue is a lack of strategy which could lead to unclear impacts and a decrease of motivation. If a company does not have standardized business processes implemented it could lead to a decrease in efficiency and slower decision making. Lastly, insufficient requirements that are not based off intelligent, thought-out decisions could eventually lead to a company’s downfall. All companies are working towards innovation and globalization, but many will fail without knowledge of some of the risks they may face along the way.

The five success factors for future organizations is the last major idea the article presents. Focusing where the money is can lead an organization to success because a lot of the times it’s not about focusing on software topics, but it’s about focusing on a few crucial business principles. Selling value rather than features is another key that should be in the head of companies. It is way more important to provide a customer with something simple they will value rather than offering something with more complex features, but less value. Steve Jobs is an example of this because he made it his mission to provide simplicity within his products. Managing relevant stakeholders is important because creating a strong core team within your main departments will increase the chance of your product becoming successful. Mastering the life-cycle can ensure governance in your company, which is something many businesses need. Lastly, continually improve. One of the most common reasons companies fail is because of their insufficient change management. I loved this article because I feel like I can take a lot about what they said and implement it into my own company one day.

Jacob Steckel

CSC402

Article Review 18

Problem Programmers

This article goes into what they call “problem programmers”, which essentially are programmers that are “bad” in a variety of ways. There are problem programmers working in companies all over the country, causing issues for corporations. First, the article reviews a few studies that attempt to detect some of these problem programmers and see how common they are. Next, the warning signs are given that detect whether a person is at risk of being a problem programmer. The author then gives his reasoning on why he believes cutting your losses is the best way to deal with these disruptive programmers.

In a study done by H. Sackman, W.J. Erikson, and EE Grant, they found that different developers could have differences of more than 20 to 1 in relation to the time required for them to debug the same problem. 20 to 1 is a huge margin! Keep in mind that all the programmers that were tested had a minimum seven years of professional programming experience. A study conducted by Tom Demarco and Timothy Lister tested 166 programmers and asked them all to complete the same assignment. The results showed that the productivity differences between programmers were about 5 to 1 on the same small project. In addition, 13 (almost 10%) of the programmers were unable to complete the project at all. Much of the research indicated that problem programmers were either unable or unwilling to follow project coding standards. The programmers negatively impacted others as well because they wouldn’t remove the bugs in their code before collaborating with others.

Companies should keep their eyes open for a few key warning signs to detect problem programmers. Rather than learning from their peers, problem programmers will attempt to cover up their ignorance. Problem programmers are territorial, they don’t like others messing or helping them with their code, even if that means delaying the project. They often disagree with final team decisions and they will bring up old topics that were moved on from long ago. These warning signs should be looked for when evaluating employees.

The article concludes by encouraging employers to simply cut their losses in regards to problem programmers. The problem with the programmers is almost always due to their attitude rather than lack of skill. Unfortunately, attitudes are much more difficult to improve than skill. The longer the problem programmer is kept around, the more legitimacy the person will gain in the eyes of their co-workers. If you see that they are a problem, do not allow the cancer to spread further. Lastly, many managers claim that they do not regret firing an employee, they just regret not firing them earlier. This article was an interesting read and it made me promise myself that I will never become a problem programmer.

Jacob Steckel

CSC402

Article Review 19

Offshore Outsourcing: The Risk Keeping Mum

This article is about how software development companies are resorting to offshore outsourcing at an increasing rate. It begins by touching up on a few of the reasons that companies are benefitting from offshore outsourcing. After, the author describes the risks, the main one being the mum effect. The rest of the article includes a study that displays five measures of cultural differences. The study is performed on the top six offshoring locations.

Offshore outsourcing is a somewhat newer idea for software developments companies, and it is growing in popularity due to its clear benefits. A couple of the major benefits include continuous operation at a less expensive cost. The top six locations for offshore outsourcing are all Asian countries. Included is Thailand, China, India, Malaysia, Philippines, and Singapore. The cost, on average, for programming in Asia is approximately 5-12 times lower when compared to Western countries. This cost is expected to remain at least four times lower in 2015 as well. So, there are certainly benefits to offshore outsourcing, but what about the risks?

There is one main concern for companies that pursue offshore outsourcing, the mum effect. This is when one or more stakeholders are aware that a project is failing but they decide to stay silent and allow the project to continue. The mum effect has led to the failure of a few multimillion-dollar software projects. One example of this is the CONFIRM project, which turned out to be a disaster worth about $125 million. After further review, it was found that the management team purposefully covered up several major technical and performance issues, and the auditors who were aware chose not to speak up. The mum effect has left some companies in shambles.

A man named Hofstede decided to perform a study to provide some information on how the mum effect occurs. He developed five measures of cultural differences and evaluated the top six offshore sourcing countries on a scale of 0-100 (including a couple scores higher than 100). Power Distance Index (PDI) measures the power inequity between people. Individualism (IDV) measures the strength of ties between people. Uncertainty Avoidance (UAI) measures the tolerance level regarding uncertainties in life. Long-Term Orientation (LTO) measures the value of long-term commitments. Masculinity (MAS) measures the clarity of social gender roles. For PDI, Malaysia had the highest rating at 104, while Thailand had the lowest rating of 64. For IDV, China, Singapore, and Thailand all had the lowest rating of 20, while the highest rating was India at 48. For MAS, Thailand had the lowest rating of 34 and China had the highest rating of 66. For UAI, Singapore had an extremely low rating of 8 and Thailand had the highest rating of 64. Lastly, for LTO the Philippines had the lowest rating of 19 and China had an incredibly high rating of 118. MAS and UAI were rated the most similarly when comparing the Asian countries and the United States. This research could be utilized to measure the risks of each country in regards to offshore outsourcing. Seeing the major cultural differences between different countries and the potential risk that comes with them was the most interesting part of the article for me.

Jacob Steckel

CSC402

Article Review 20

Risk Profile

The name of this article is “A Risk Profile of Offshore –Outsourced Development Projects” and it is based on a study that ranks 25 risk factors in regards to offshore outsourcing. This article claims that offshore outsourcing may not be worth it in the end due to the risks involved. After listing the 25 risk factors and rating them based on their significance, the article dives deeper and provides a description on each of the top ten risks.

Per Forrester Research, 65% of European and American enterprises currently utilize offshore providers for application development. An additional 13% of enterprises say they are planning to begin within the next year. In comparison, two years ago only 45% of enterprises used offshore providers for their development. So, the increase is clear, but a recent ACM report claims that outsourcing “magnifies existing risks and creates new threats”. Thus, organizations will be required to allocate additional funds and resources to deal with these new risks and threats. Is offshore outsourcing worth all the additional risks that comes with it?

The article provides 25 risk factors, but I am just going to stick with the top ten since they are the most meaningful. Ranking at number one is the lack of top management commitment. Top management is essential to projects and without it a few challenges that could arise include political battles, delays, or even rejection. Rank two: Original set of requirements is miscommunicated. A lack of face-to-face communication between developers could lead to lack of understanding of requirements. Rank three: Language barriers in project communications. This one is self-explanatory, communications could become increasingly difficult and frustrating if developers are unable to speak the same language. Rank four: Inadequate user involvement. If you cannot monitor employees in-person, user participation could become a significant problem. A decrease in user involvement will also hurt efficiency. Rank five: Lack of offshore project management know-how by client. Many companies are new to offshore outsourcing and are not prepared to send in-house experts to monitor the offshore project. Rank six: Failure to manage end user expectations. Since the users are not in direct contact with the developers it will be harder to manage their expectations for the project. Rank seven: Poor change controls. If an organization needs to make changes to the initial set of requirements, it may be harder to get the message across when the developers are overseas. Rank eight: Lack of business know-how by offshore team. Overseas resources tend not to have a strong understanding of a client’s business context. Rank nine: Lack of required technical know-how by the offshore team. It is difficult to ensure that the developers overseas have the quality technologies that are required for the project to go as planned. Rank ten: Failure to consider all costs. When offshore outsourcing, many companies fail to account for all the hidden costs that are included.

Personally, all those risks would stress me out and I’m not sure if offshore outsourcing would be worth it. As its popularity increases, organizations will have to decide for themselves if they feel that the benefits are worth all the risks.