Egypt–SPIN Newsletter

Issue 17, Jan. – Jul., 2008

Sponsored by SECC

From the Editor ***(Ahmed S. El-Shikh)***

Welcome to our 17th issue of Egypt –SPIN newsletter. In each issue we try to put together relevant information in the form of articles and recaps covering the previous seven months events hoping to provide our members of Egypt – SPIN with information to support their current interests.

We are used to announce the major events in the field of process improvement in Egypt. At the same time, it is a pleasure to announce many success stories and great achievements that happened during the previous period of time.

Please, refer to SECC website, www.secc.org.eg, for more information. Hereunder, you can find a summary list of the most important events:

- **January to July 2008**
  - **SECC** Participated in the **SEI SEPG Europe 2008,** conducted during **June in Munich, Germany** with **two sessions.**
  - **SECC** signed a cooperation agreement with the **European Software Institute (ESI)** to establish a **Software Engineering Center of Excellence** in Egypt referred to as **"ESICenter SECC"**.
  - **SECC** released a **specialized journal in software engineering** in January 2008 entitled **"International Journal of Software Engineering, IJSE"**. IJSE Provides forum for the state of-the-art research in different fields of software engineering.
  - **SECC** participated in **GITEX Riyadh 2008**.
  - **SECC Event** organized by **ARCOM** in **Saudi Arabia** for **promoting CMMI**. The event was under the patronage of his Excellency **Dr. Mohammad Alsuwaiyel**, president of King Abdalaziz City for Science & Technology, and with contribution from **Microsoft Arabia**.
  - **SECC** started the "**CMMI 2008**" project with **6 Egyptian companies** in Egypt to achieve **CMMI ML 2 and 3**.
  - **SECC** started the "**SPI for SMEs 2008**" project with **20 SME** companies in Egypt to apply the "**SPIG Product Suite v1.1**".
  - **SECC** started the "**TSP CMMI 2008**" project with **3 Egyptian companies** in Egypt to achieve **CMMI ML 3**.
  - An international consulting firm has been awarded the bid of "**Consultation on Establishing a Software Testing Center (STC)**".
  - Two **SECC Quality Experts** were qualified as an **authorized SCAMPI Lead Appraiser** from the **SEI**.
  - Two **Egyptian companies** achieved **CMMI ML 2**; the SCAMPI A was led by two **Egyptian SEI-Authorized SCAMPI Lead Appraisals**.
  - Two **SEI courses** entitled "**Introduction to CMMI**" were conducted by two **Egyptian SEI-Authorized CMMI Instructors**.
  - **ITIDA/ SECC** Sponsored **50 software companies** to attend a **two-day conference** at Sharm El Sheikh on June 18th – 19th on "**IBM Rational Software Development**".
  - **SECC** organized an **Embedded Software Training Track** at the Smart Village.

This issue introduces some hot topics in two series and three independent articles as follows: CMMI ML2 Extensions to ML3 (1st article), Critical Success Factors for SPI in SMEs (2nd article), Risk Management Framework (3rd article), Testing Tools and Automation (4th article) and Sound Software Engineering Practices (5th article).

Eng. Ahmed Abd El Aziz *completes his series that raises a warning flag to software companies that are ambitious to achieve the* ***transition from CMMI ML2 to ML3 using traditional approaches in only one year****. In addition, he summarizes the new required* ***Extensions of ML2 Practices****.*

Eng. Ahmed Mahdy *summarizes his experiences in the field of* ***Software Process Improvement in SMEs*** *including the* ***most critical success factors****.*

Eng. Ahmed Gad *summarizes his experiences in the field of* ***Organizational Risk Management*** *and* ***Organizational Security****. His article discusses the needed theoretical primer for understanding basic security concepts for IT Organizations.*

Eng. Omar Kamal *explains a practical approach for* ***SW test automating****. The article present a tool that the writer has been using for a while and it shows great benefit in* ***automating and enhancing test design process****.*

Eng. Mostafa Hamza *explains his opinion on the* ***importance of sound software engineering practices****. He gives a brief overview on* ***SEI Personal Software Process (PSP)****.*

We hope we succeed to give you an idea about what is going in our community. Please write to the editor your comments about our progress. We always ask you to submit short articles for publication that deal with your experience in defining, developing and managing software efforts as well as process improvement experience. Remember that our goal is to encourage an interchange between our readers. You can email spin@secc.org.eg or aselshikh@mcit.gov.eg

Table of Contents

From CMMI ML2 to ML3 in One Year! Part3: New Practices

By: Ahmed Abd El Aziz

In the previous two articles we mentioned the difference between ML2 and ML3 from two perspectives; the expectations of ML2 process areas when appraised against ML3, and the extension or elaboration of some practices when moving from ML2 to ML3.

This article will focus on the new practices that were not mentioned at all at ML2. This does not necessarily means that they are not implemented in ML2 organizations. This only means that they are not checked during ML2 appraisal.

Handle Process Improvement as a Project

In ML2 there are lots process improvement activities. These activities may be formally planned and monitored and may be not. This is never checked during the appraisal. When it comes to ML3, the process improvement activities themselves must be handled as a project that has a set of requirements, a plan to implement these requirements, and a mechanism to monitor implementation of this plan. The following points have to be addressed:

- What are the organizational needs from the process improvements? What areas need to be improved? What are the priorities of these improvements?
- How and how often will the organization appraise its internal process to identify what is good and what still needs to be improved?
- Preparing an action plan identifying the SEPG Team, roles and responsibilities, steps to define and review the processes …
- Preparing a deployment plan for the processes and implementing it. Example of the issues addressed by deployment plan; if a new template is prepared for project schedule, shall all projects use it, only new projects shall use and running projects will ignore it, or running projects will partially use parts of the new template?
- The plan has to be monitored to make sure it is followed. Plan is not created for the sake of planning only, it is created to be implemented and followed.
- There must be some mechanism to learn from the deployment of the process. Even good written and reviewed processes have a lot to say when implemented or piloted.

Analyze the Requirements

One of the concepts behind the creation of CMMI is that most of the problems of the software projects are actually project management problems, not engineering problems. There are talented software engineers who can develop the product. The

problem is that the companies fail to deliver the required product on time, within budget, and meeting the required quality. That is why ML2 focused on project management and ignored – to some extent – software engineering.

In ML3 engineering comes into focus. Basic project management practices are in place and it is time now to pay attention to analysis, design, coding, integration, and testing.
Starting with the requirements, the requirements must be analyzed. To be more precise, the following points must be addressed:

- What are the business components of the system to be developed?
- What are the requirements of each components and how these requirements are derived from the customer requirements of the system as a whole?
- How these components interact with each other and with the external world?
- What are the operational scenarios of the system? These could be represented as use cases, timeline scenarios …
- Technically review the requirements to make sure they are necessary, sufficient, consistent, cover stakeholder needs …

Develop Alternative Designs and Select the Best

This is one of the practices I like much in CMMI. It enforces staff to think in alternatives, not to just accept the first one comes in mind. When designing a solution, this practice directs the design team to think in possible multiple solutions, evaluate them, and select the best of them. In many cases people briefly discuss one solution – most of the times they not discuss, someone of the seniors just takes the decision – that will impact the work of many years without giving few to time to ask themselves "Is not there any other way to do it better?". Implementing this solution many times before does not necessarily mean it is the best solution – even it was the best solution in the past.

Develop Detailed Design for Components

In the design phase, the system is usually divided into components (could be called sub-systems, modules, packages …). The architecture shows these components and their relations and interactions. The team has to discuss for every component is it better to reuse it from another project, buy it from some third party, or build it as part of the project? If the team decided to build it, they must create detailed design for the component. The word "detailed design" does not enforce using some specific design approach or technique or some specific level of details. This should be identified based on the project and organization needs and nature of the project.

Implement the Design

At the end of the day the design must be used to build the system, not just to prepare documents to satisfy the appraisal team and ignore them when writing the code. This practice directs us to consider the level of detail of the design mentioned in the previous practice. For the design to be used it must be well understood and suitable for the developers. Sometimes very detailed design is required and the

developers has to follow it systematically, while sometimes the design could be to a higher level of details leaving freedom to developers to take many decisions during coding. In all cases the design must be followed and there must be some mechanism – most probably technical review of the code – in the organization to make sure the design is followed.

Develop Supporting Documentations

Many systems are complicated and need some sort of accompanying documents to be delivered to the customer with the system to ease or explain how to install, use, and maintain the system. Examples of these supporting documentations are:

- End-user training materials.
- User's manual.
- Operator's manual.
- Maintenance manual.
- Online help.

Integrate the Product

As the developed system – especially large ones – are composed of many components, the project team has to think how these components will be integrated together and delivered to the customer. They have to address the following points:

- What is the integration sequence? In other words what are the components to be integrated first then what is next? For example if we have a multi tear system that is divided horizontally into layers and vertically into modules, shall it be integrated by layer (e.g. lower layers first then higher layer), by module (e.g. modules that offering services to other modules first then modules that are using the services), or mix of the two approaches?

- What is the integration procedure? For example if we are testing the integration of three modules while one or more of them depends on the services provided by other module that is not ready yet for integration, shall we create a stub (dummy module that provides dummy services) to perform the test? If we decided to create the stub, shall we test the stub to make sure it really simulates the missing module?

- Control changes to the interfaces between the modules – which are defined earlier during the design – and handle any changes to them formally as you handle the changes in the requirements (e.g. via formal change request).

- How the solution will be assembled and delivered to the customer? Shall we burn a CD and deliver it with installation instructions? Shall we create a setup application? Shall we upload some files to the customer's web server?

Technically Review Major Project Documents and Analyze the Results

During the lifecycle of the project, many documents are generated like the project planning (estimation, schedule, QA plan, CM plan …), analysis, design (architecture, detailed component design, database design …) and testing documents (test plan, test cases …). Which of these documents need to be reviewed and how this

review will be performed? There are many review techniques ranging from simple superficial techniques like walkthroughs to formal techniques like formal inspections. The organization has to decide which documents to review and which technique to use with each document.

After reviewing the work products, the review result has to be analyzed to identify suitable corrective and preventive actions to address the defects discovered during the review. Examples of this analysis include:

- Phase injection (which phase injects defects) and phase detection (in which phase the defect was detected.
- Number of defects discovered compared to number of defects expected.
- Severity of defects discovered (e.g. % of major, normal, and minor defects)
- Causes of defects (e.g. lack of experience, lack of understanding of the business domain, use of inappropriate template or process …)

Test the Product and Analyze Testing Results

The product must be tested before being delivered to the customer. This does not necessarily means it must be tested within the organization. It could be tested in an external testing center not belonging to the organization.

CMMI does not require or identify what tests to be performed (e.g. unit test, system test …). This depends on the nature of the projects and the needs of the organization. For example stress testing is needed in many web applications while it is not needed in most of desktop applications.

After testing the test results has to be analyzed to identify suitable corrective and preventive actions to address the defects discovered during the testing. Analysis techniques similar to techniques mentioned above in analyzing technical document review results could be used.

Perform Acceptance Test and Analyze the Results

If the system is tested and it successfully passed the test, does this mean that it will work correctly at the end-user environment? Do not we need to test it on the end-user environment to make sure it really works?

In many cases the system works very well during development and testing, but when delivered to customer many unexpected problems appear. These problems must be detected and fixed before final delivery to the customer.
To detect these problems the system has to be installed on the end-user environment and tested on that environment. This test could be done completely by the end-user, by the testing team, or by mix of both.

By analyzing these problems we may discover the root causes of many problems that we have to avoid in upcoming deliveries either to this customer, another customer, or even the delivery of a completely different system. For examples the cause of some problems is that there are some special settings in the development and testing environment which are different from the settings on the customer's.

One way to mitigate the risk of having problems during acceptance test is to prepare a testing environment that is

much similar to the customer's. Sometimes it is not easy to perform acceptance test. In this case testing on a similar environment is accepted as an alternative practice to acceptance testing.

Follow Formal Mechanism When Taking Critical Decision

In every day a lot of decisions are taken either in the projects or in the organization. Some are very critical – like changing the organizational structure – and some could be trivial – like implementing a loop with a "for" or "while" statement. For the critical decisions there must be some formal mechanism to take the decision.

Changing the organizational structure is not a simple decision to be taken without an extensive study addressing the advantages and disadvantages of the proposed structures while deciding which statement to use to implement a loop could be completely left to the developer to decide without even having to justify why he preferred one statement to the other. The following points have to be addressed:

- The organization must have some guidelines that identify what needs a formal decision taking mechanism and what does not. Example of areas that may need formal decision are:
    a. Selecting lifecycle to follow in the project.
    b. Selecting technical solution to implement in the project.
    c. Performing Make/Buy/Reuse analysis of project components.
    d. Selecting supplier to deal with.
    e. Selecting training provider and technique to use to conduct training.
    f. Identifying the criteria based on which the alternatives will be evaluated.
    g. Identifying the possible alternative solutions.
    h. Evaluating the solutions using the criteria and selecting the best of them.

Conclusion

After reading this article and the previous two articles in this series, the management has to answer the following question "Is it the right decision to go directly from ML1 to ML3 in one year?". From consultation point of view the magic answer is "It depends!"

Biography

Ahmed Abd El Aziz has over 11 years of experience in the software industry. During that period he worked as developer, project manager, department manager, and process improvement leader. He joined SECC in September 2006 as a Senior Quality Consultant. During that period he offered consultation on CMMI to many software development and implementation companies to maturity levels two and three. Ahmed is certified PMP since May 2005 and is SEI-Authorized SCAMPI Lead Appraiser since March 2008.

Feedback Contacts

Feedback, comments and questions are appreciated by the author.

Email:
aaziz@mcit.gov.eg

Critical Factors for a Successful SME-SPI

By: Ahmed Mahdy

"The improvement of process is the blood of quality", confidently my article can start with this quote; Software Process Improvement (SPI) is the blood of software quality in any IT organization.

Knowingly, Software Engineering Competences Center (SECC) has performed the Small Medium Enterprises (SME) project for SPI with 20 Egyptian companies, and generally I was pleasured by my participation with this project and specifically the consulting period, my being dedicated to this project to implement in my company which has multi-conceptual directions in Software Engineering, has reported some feedback (which was reported to SECC for kind of constructive improvement) and factors (learned lessons) for a successful project that I would like to line them here:

Factor-1: Management adoption

The serious adoption of senior management should be active during the whole period of this project, and you can ask "What can the manager do in view of this situation and this factor?" Glad you asked. The manager should delegate a responsible for this project and consequently he requires a periodic progress meeting with a report (I recommend a weekly report, however it's subjective), the manager has to take the obstacles or presented issues on a high level of importance and try to handle each.

- For the first part in the project (the training): the manager selects the suitable persons each one in his/her area, and try to find the best one who can transfer this knowledge after that, and I prefer to ask each one to prepare for holding a presentation about the course s/he will take as soon as s/he returns to the company.
- For the second part in the project (the consulting): the manager opens a channel for following up with the consultant parallel with the settled channel with the responsible of this project inside the company.

Factor-2: SPI guide is an initial model but it will never be your quality system

Do not stare at SPI guide only in the definition level, but also take into your account carefully and constructively at least the following:

- Your company policies
- Work Environment
- The rigid constraints
- All project types (including short-term future projects)
- Current level of development in your human resources
- Projects history

You can gain the information for the points above (or more) by interviews, meetings, and open discussions with the empowered stakeholders. Nevertheless, you may not have the clear evidences to gain such information and if you have any conflict or alternatives, you can escalate to the senior manager to resolve that after feeding him/her with the gained information for a better decision support.

More importantly, focused discussions with the project managers, team leaders or technical managers should be activated during the definition level, and you should propose some justification (or rational behind) for any change in your process structure, content, …etc (and the evidence can be easily the MOM –Minutes Of Meeting- with stakeholders who you discussed with), thereby, you can refer to that in case of any other process change or human resources change.

Factor-3: Definition level is the prerequisite of the first step

Although the definition period is very important in this project, this period is a pre-initial step for the improvement level. In the definition, you should focus on what you really currently have (i.e. raise the AS-IS model) but do not go through defining many new ideas or thoughts (i.e. if you have TO-BE model to do, I recommend not to include it all in this level of definition).More effectively, never leave any word in your definition that you do not understand its (How, When and Why) and always keep these questions in your mind while you are defining.

Factor-4: Project success is everyone's responsibility

This factor is easy to understand and hard to apply but not impossible. Surprisingly, you will not be able to perform the definition without other stakeholders' participation, and accordingly, you can use this level to attract other resources in the participation constructively by taking their opinions in the processes' definition and other related decisions and taking into account the senior management approval, and just make this level to break the ices.

Unfortunately, I cannot claim that you will not find any resistance in this change but I sign on the opposite. Yes, you will find a resistance in your way and the right question to ask is "What should I do towards this resistance?" all the combination of these nine factors will give you some insights in dealing with such cases.

Factor-5: During this project, avoid useless and conceptual conflicts with other models

In this project, you may find the opportunity for researching and software quality discussions which may open other models and theories. Here, you consult the SECC representative if you want to include any, however, I do not recommend throwing out any thoughts, ideas, models or even theories that you find out (i.e. record them somewhere where you can get back to after this project) AND do not recommend using them in this level, only go with this project until the end.

Factor-6: Keep your organization updated with other models

This factor does not necessarily conflict with the one above; because keeping yourself updated with other ideas and models will help you in a more suitable definition in your organization which leads to a more agility, flexibility, accuracy, measurability, or stability in work and delivery, just keep the findings on your researching wallet.

Factor-7: Do not fabricate the action items

It's easy to fabricate the required action items (processes, work products, tasks…etc) but it's really time-wasting and hard to handle its

negative effects on your quality management system.

Factor-8: Return On Investment (ROI) measurement

In fact, you should not expect that much to gain the fruits in this project's season, however you have to keep the ROI reflected and considered in your decisions (how?) at least: name an internal project for the training period, or divide it into multi-projects if your measurement needs that, and this approach works with any project like SPI one, again expect a negative value in the period of training and consulting which will be overthrown and toppled by your implementations (mostly, after the first 2-3 implementations as long as your improvement is working fine), thereby the measurements of ROI is about professional reporting to the management, precise evidence of your development and improvement.

Factor-9: Project responsible (representative) and some characteristics

This project has to be assigned to a team or a trustful person who should have the following (from my experience):

- Excellent language that the organization uses.
- Wide knowledge of the organization environment
- Open minded and ready to apply what is coming better even if he will change his previous decisions
- Excellent Communication skills, very important to deliver the concept to others
- Long breath and patience is important for facing the very expected resistance
- Ready to learn and read in a constructive flexible way
- Good presentation skills
- High commitment in time and tasks for achieving even more than the required action items
- Self-motivated and self-confident

Eventually, I advise the responsible of this project to have my quotes that I have derived after my ownership of similar projects "Change does not change" and "Face the resistance of change as a FACT not as a problem"

Biography

Ahmed Mahdy is a decision support and software engineer, graduated in 2006 from faculty of computers and information, Cairo University. His main interest areas are Software Engineering, Software Process Improvement, Process Modeling and Simulation with proven track record in programming using many languages. He has been working in modeling and implementing workflows, since 2004 with emphasis on the e-government projects. He has six years experience in Software Engineering working beside his academic study. He participated in international software engineering conferences with scientific papers sponsored by IEEE.

Feedback Contacts

Feedback, comments and questions are appreciated by the author.

Email:
amahdy@sahmgroup.com

Risk Management Framework Part I

By: Ahmed Gad

Introduction

In previous newsletters, we discussed basic security concepts; Availability, Confidentiality, Integrity, and Assurance; and How they can be implemented in Organization Infrastructure through many safeguards (Authentication Techniques, Encryption, ACL …etc).

I named this series "The definitive series" which provides a needed theoretical primer for understanding basic security concepts. Actually, not only these safeguards are used for achieving Security cornerstones, but also many other safeguards are used like Firewalls, Intrusion prevention, Antivirus programs, Host Intrusion preventions …etc. All these safeguards should be complemented with a proper security policy, Security plans (incident handling, business continuity and Disaster Recover …etc), Well-defined Organization roles and responsibilities, and well defined processes for change and configuration management.

Okay, A lot of work is needed to achieve a proper security system ended with a security information management system which is capable of not only reducing the costs by incorporating security in Organization culture but also by adding value to the customer. As we finished the definitive phase newsletters, we'll move to the second phase which is Security planning.

The first step in Security planning is how we can identify and react to the probable risks to the organization assets which provide the needed business profits to the organization. In this article, we'll explore the systematic approach to risk management framework.

Risk Management Framework Approach

In the real world, needs must be balanced and limited resources allocated to a select set of security safeguards to protect the organization assets. Risk management is the practice of balancing those needs and selecting those safeguards.

Managing information security risks is a systematic process run as follows:

- Writing the security policy (with business input)
- Creating Assets Inventory and Determining the value of information assets and infrastructure
- Identifying and itemizing the threats to those assets and infrastructure, Estimating the likelihood that a threat will be realized, and Calculating the total cost of threats
- Analyzing Assets Vulnerabilities to those threats
- Analyzing risks, or identifying industry practice for due care
- Devising and implementing mitigation strategies to minimize risks while improving

education, forensics, feedback, process reviews, and other elements of security management

- Designing controls (e.g. Implement intrusion detection and incident response), and writing standards for each technology
- Deciding what resources are available, prioritizing countermeasures, and implementing top priority countermeasures you can afford
- Setting up a security infrastructure
- Conducting periodic reviews and possibly tests

We'll postpone discussing Security policy (for this time) as it has a considerable business perspective. Instead, we'll start discussing the technical steps of risk management, how they are implemented, and the tips of performing in a good way.

Inventorying and valuing Information assets

The first step to understand risk is to know what should be protected. The value of information will vary based on the type of information, the age of the information, and the cost of re-creating it.

For example, consider a manufacturer's database of thousands of active customer records. The database includes purchases, invoices, pending orders, works in progress, and plans for custom parts designed to customer specification. To determine the value of this information, the company would need to consider:

- The cost of re-creating the database from electronic backup or paper records
- The permanent loss of this information
- The value of information assets
- The value of the information to a competitor
- Business disruption
- Exposure to additional vulnerabilities that a defective state will create
- Effect on brand and customer goodwill
- The liability associated with unauthorized disclosure or delays in shipments
- The cost-delayed order fulfillment and billing, and the subsequent effect on cash flow
- The cost of labor of workers who cannot do their jobs while the system is down
- The likelihood that customers would switch to a competitor because of the effects of the Incident

There are certainly other factors, but this short list demonstrates that the cost of a failure includes more than the cost of hardware and the IT staff's time to recover.

The true costs of failure ripple through the organization sometimes quickly—as in the case of lost staff productivity—and sometimes slowly—as in the case of impact on cash flow and customer loyalty. Your management knows how much they pay people. They know what hardware and software purchase, maintenance

and licensing costs are. But they may not be aware of the value of information assets. It may be worth your time to carefully document the valuation of the assets you wish to protect with your increased countermeasures. Needless to say, one of the most important valuations is revenue. Anything that is central to a revenue center will be likely to receive support as opposed to overhead devices.

For example, governments and research-based businesses are more likely to suffer from information theft than commodity manufacturers. The intellectual property of a pharmaceutical company can represent millions of dollars in Research and Development (R&D) investment; the processes used by a textile manufacturer may be of interest to a competitor but also easily independently developed.

The information assets may be hardware to host information, Commercial Software to serve the customer, Databases holding the customer and purchases data, human resources working as operation team, intellectual property …etc. You may categorize all these resources into three categories:

- Material (Data, information, Knowledge)
- Equipment (Servers, Network, Workstations, physical site preparation)
- Human (Employees)

Maintaining assets of all these information, their value, and their impact on business in case of loss is a good start to Risk management.
Not only you have to maintain all these information, but also updating the information inventory Database continuously is a challenged consideration that should be undertaken by the organization.

There are a lot of Network/ Security management software that enables you to do this task in a proper and easy way. It scans the Network in no time to come up with an exact assets inventory Database. The management software, however, may need some metrics from your side to complete the database. These metrics, which are subjective, are to measure the value of corresponding assets which may differ from one organization to another.

Threat Calculations

A lot of questions are coming here, what are we afraid of? What is it - can we name it specifically or is it just a vague, uneasy feeling?

If the threat is successful, how bad will it hurt? What is the probable extent of the damage?

How often is this likely to occur? We are more willing to accept the risk of a threat that is not likely to happen often. But, if something can damage us on a daily basis, this is a significant problem.

Finally, how do we know? In the cyber world, how accurate are our risk calculations when new program or operating system vulnerabilities are discovered weekly?

For example, if you run a DNS server that has known vulnerabilities and is neither patched nor shielded by the perimeter, it is certainly going to be compromised. It might not happen in a single day, but it will happen over the course of a year. Well, once they compromise the DNS box they have the ability to manipulate the addresses associated with the names of the network entities (such as computers) at your site. These names and

addresses are often used to identify which computers are allowed to access other computers – which is your organization's trust model. If you have valuable assets, that may be what happens. Or they may just create weird system domains and hit systems all over the Internet, giving your organization a bad name. They will compromise the DNS server, most likely via a buffer overflow. How bad would it be? If they chose to manipulate the trust model and had several days to work without being detected - such as over the Christmas holidays – they could make considerable headway at owning the entire organization's information assets.

Threats are persons or processes that can exploit vulnerabilities to breach or damage information systems. Well-known threats include viruses, worms, DOS attacks, information theft, power failure, fire, and flood. SQL Slammer, which struck on January 25, 2003, effectively shut down large segments of the Internet. Email systems and other collaboration tools have been plagued with blended threats that combine mutating viruses, worm techniques, backdoors, and other attacks. MSBlaster, SoBig.F, and Lovsan are just a few of the blended threats that have made the news.

The threat vector model can help the people analyze the types of threats. The probable threat vectors are indicated as:

- Outsider attack from internet
- Outsider attack from telephone
- Insider attack from local network/ systems
- Other indirect threats types

Outsider Attack from internet

The probable sources may be:

- Passive attacks by knowing the organization's domain details
- Newspaper, web articles on attacks at other places, if it happens to them…
- Hacking web sites: www.antionline.com, www.sobotage.org
- Firewall/Intrusion Detection logs are an excellent source for specific threats
- System audit trail logs are as well
- Demo an intrusion detection system
- Application security vulnerabilities
- E-mail attachments

Following are two examples illustrating how you can identify network problems:

Example 1: If you scan your site for SNMP agents that answer to the community string "public" you may be surprised how many there are. Then you can look in the IP MIB for routes that are cheaper than the one through your own firewall. These could indicate a back door. Can an intrusion detection system find evidence of back doors? Sure, I have found several when I see lost packets, especially broadcast packets, from our internal address space coming in from the Internet trying to find their way home. You can do the same thing with your filtering router or firewall of course.

Example 2: Many wireless networks are installed with default settings and can be an enormous hole in your defenses. Try connecting to your network with a wireless device from the parking lot or from the building

across the street. If you can do it, so can your competition!

Outsider Attack from the phone

The probable sources may be:

- IP telephony attacks
- Unintended/ intended Information disclosure by telephone conversations with the employees
- Commercial IP phone SW (e.g. www.sandstorm.net, Net2Phone, SKYPE …etc)
- Instant messaging and chatting
- Collaborative applications (Net meetings, interactive conferencing applications …etc)
- VPN remote access

Insider Attack from local network/ Systems

The probable sources may be:

- Running malicious SW (e.g. port scan, TCPwrappers or Psionic log and Port Sentry reporting to central SYS logs, Keystrock monitor)
- Installing unlicensed/ unwanted SW
- Document loss (disclosing classified information)
- Removable media handling
- Physical access to Data Centers, Servers, Workstations …etc
- Stealing the central LOG

Other indirect Threat Types

Other probable threat types may be:

- Physical access
- Power interruptions and unstable voltage
- Temperature changes
- Flooding and water
- Sabotage and fishing
- Human resources turnover
- Factors affecting Equipment depreciation rate
- System Administration Comfortable Environment
- Misunderstood Systems configuration
- Unstable internet connections
- Contractual risks

Note: Many threat types have considerable effort from security professionals who devoted best practice effort to come up with a consensus threat types' model. Many organizations are availing such information (e.g. SANS Research Consensus Projects, Center for Internet Security)

After identifying the threats, we come to calculating the probability of the threat occurrence. Estimating the likelihood of threats is probably the single most difficult step in risk analysis. In the best cases, organizations have enough data about past threat activity to reasonably predict future occurrences. For example, audit logs from network monitors can reveal probing by hackers looking for vulnerable ports.

In other cases, risk analysts can estimate probabilities based on public information, such as virus-threat levels published by antivirus solution developers and the Computer Emergency Response Team (CERT) at Carnegie Mellon University. In yet other cases, such as the threat of a contract programmer placing malicious code in an application, exact quantitative estimates are not possible. For more information about CERT, threat indexes, vulnerabilities, and basic security statistics, check out http://www.cert.org/nav/index_main.html.

The likelihood of each threat is estimated on a scale of 0.0 to 1.0 where 0.0 means that the threat will never occur and 1.0 means the threat will certainly occur.

With the value of assets and the list of threats and their likelihood in place, the next step is calculating the total cost of threats.

Calculating the cost of risk is not an easy task, we'll simply explore the basics. Risk analysts calculate the total cost of a threat in a three-step process.

- Estimate the exposure factor or the value of a loss if a threat is realized against an asset. For example, if a fire destroys a server with a customer database valued at $250,000 and all but 10 percent of the database can be restored from backup, then the exposure factor is 10 percent.

- Calculate the single loss exposure (SLE), which is simply the asset value multiplied by the loss exposure. In the customer database example, the calculation is $250,000 × 10 percent, which equals $25,000.

- Calculate the annualized loss exposure (ALE), which is the number of times a threat will occur in one year multiplied by the SLE. Assuming a fire occurs once in 10 years, the ALE of the fire example is 0.1 × $25,000, which equals $2500. The ALE is the maximum amount an organization should rationally spend to protect against a threat, at least in theory.

More examples

- A company's top salesman accounts for 25% of their $40 million in revenue, or $10 million. His client contact list and fee schedule is stored on his laptop and is not encrypted. If it fell into the wrong hands it would be worth at least 10% of its value to the competition ($1 million) and possibly more if they can finesse the information. So we find we can calculate a minimum approximate SLE, but there is uncertainty as to a maximum value.

- SLE: 1000 employees, 25% waste an hour per week surfing, $50/hr x 250 (25% X 1000) = $12,500. ALE: They do it every week except when on vacation: $12,500 x 50 = $625,000

For most applications the best approach is the financial one, with the exceptions of critical systems (such as nuclear plant control) and weapon systems. However, it does take a lot more effort to quantify what the value of things is, and so the qualitative approach is far more popular.

The single biggest problem with the qualitative approach is in the implementation - people tend to mark

"low risk" even if it is other than that. Or they mark "medium" or "high" for their pet peeves as opposed to actually calculating the risk.

The differences between the two approaches are:

- Qualitative is easier to calculate, but its results are more subjective
- Qualitative is much easier to accomplish
- Qualitative succeeds at identifying high risk areas
- Quantitative is far more valuable as a business decision tool since it works in metrics, usually dollars

In the upcoming newsletters, we'll continue discussing the rest of the risk management framework process.

References:

1- SANS training tracks – Risk Management Framework - www.sans.org

2- Definitive guide to security management – Dan Sullivan – Computer Associates – www.realtimepublishers.com

Biography

Ahmed Gad Al-Karim, is a security and infrastructure consultant in the Egyptian e-Gov Program. He has 7 years of experience in the field of information technology. Currently, he holds Techno-MBA. Information security is his major interest. His interests include ISO 17799, BS 7799 security systems.

Feedback Contacts

Feedback, comments and questions are appreciated by the author.

Email:
ahgad@mcit.gov.eg

Tools for N-Wise testing

By: Omar Kamal

Introduction

In SPIN Newsletter vol. 15, a powerful testing technique was introduced that can both used as a black-box or a white-box testing strategy. There exist a number of software tools in the market either free or commercial.

This article won't provide the readers with a tool survey, however it will present a tool that the writer have been using for a while and it shows great benefit in automating and enhancing test design process. The tools can be downloaded from (http://download.microsoft.com/download/f/5/5/f55484df-8494-48fa-8dbd-8c6f76cc014b/pict33.msi). Reading the previous article "Software Testing Techniques Series: Pairwise Testing Technique " published in vol.15 is a pre-requisite for understanding this article.

Pairwise Independent Combinatorial Testing (PICT) Tool

One of the most practical and easy tool for developing pairwise test cases is "Microsoft PICT", the tool offers N-Wise reduction for exhaustive test cases. Besides it simplicity, there exists a number of features that makes the tool very handy and powerful at different levels of testing (unit, integration and system level testing).

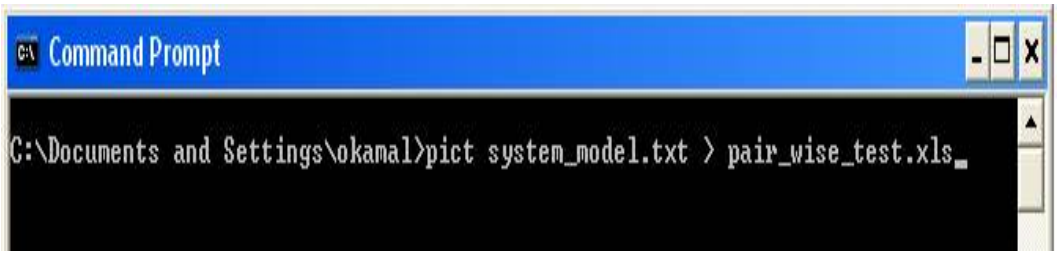
The tool input is just a text file that captures system inputs and their possible values (see figure 1).

Each of the system's input value is presented in a single line as follows:

<ParamName>:<Value1>, <Value2>, <Value3>, ...

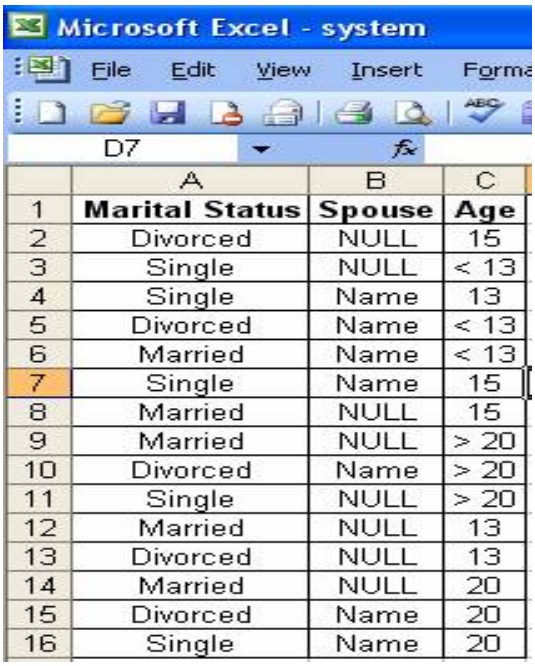
```
# -------- System Inputs -----------------------------
Marital Status: Single, Married, Divorced
Spouse: NULL, Name
Age: < 13, 13, 15, 20, > 20
```

After listing all system inputs and their associated possible values in the input ASC-II text file (Name it for example, system_model.txt), generation of the test cases is done as follow:



pict system_model.txt > <pair_wise_test.xls>

Where the "pair_wise_test.xls" is the name of the file that store test cases in Comma Separated Values format. The output will be a reduced set of test cases that satisfy the pairwise criteria



	A	B	C
1	**Marital Status**	**Spouse**	**Age**
2	Divorced	NULL	15
3	Single	NULL	< 13
4	Single	Name	13
5	Divorced	Name	< 13
6	Married	Name	< 13
7	Single	Name	15
8	Married	NULL	15
9	Married	NULL	> 20
10	Divorced	Name	> 20
11	Single	NULL	> 20
12	Married	NULL	13
13	Divorced	NULL	13
14	Married	NULL	20
15	Divorced	Name	20
16	Single	Name	20

The following section describes what are the main important features that PICT offer.

Main Feature N-Wise testing

Although the name of the tool may indicate that it is limited to pairwise reduction only, the tools offers more than pair-wise combination it offers N-Wise reduction by defining a command line argument that specifies the type of reduction the tester need. Such feature offers more strength in testing going beyond basic pair-wise algorithms. Practically, pairwise testing is enough for testing med-to-large systems, and the marginal gain of testing the overall system based on 3-wise or more testing gets lower by increasing N. To generate three-wise test cases type the following

pict system_model.txt > <pair_wise_test.xls> /o: 3

Where 3 indicates three-wise test case generation.

Model Constraints

Most of the time, the inputs for the system under testing are dependent in a certain way or another. For example, let us assume a field 'A' that describe the marital status having the following possible values "Single, Married, Divorced, etc ...", and another field 'B' that hold the spouse name having the possible values "Null/String". So it makes a lot of sense that If f1 = "Single" Then f2 must be "NULL", how can the tester constraint the PICT results to obey such rule. PICT offers a section for defining such constraints and the process of generating test cases will always respect those constraints while satisfying the N-Wise rule. For example to add such constraint in the previous example, we add the following line after variables definitions.

IF [Marital Status] = "Single" THEN [Spouse] = "NULL";

```
# -------- System Inputs ------------------------------
Marital Status: Single, Married, Divorced
Spouse: NULL, Name
Age:  < 13, 13, 15, 20, > 20

# ------------ Constraints --------------
IF [Marital Status] = "Single" THEN [Spouse] = "NULL";
```

PICT supports a number of constraining built-in keywords like (***IF, THEN, ELSE, NOT, AND, OR, IN, etc ...***). In addition, most of the conditional operators that exists in any programming language you will find it in PICT (**=, <>, >, >=, <, <=, *and* LIKE**.)

Other Features

PICT not only simple to use, but it includes a number of great features that can be rarely found in any other similar tool. The following list of features is just an example of such important and practical features:

Seeding

Allows the tester to initially feed the PICT with already implemented test cases as a seed for the PICT. Doing so, the PICT focus on generating the delta test cases that offers the missing cases needed to achieve the N-Wise criteria.

Aliasing

Allows the tester to provide different names for certain value attribute to a certain parameter. For example:

Spouse: ***Empty|NULL****, Name*

Here "Empty" and "NULL" represent the same value name. Multiple names do not change the combinatorial complexity of the model. No matter how many names a value has, it is treated as one entity. The only difference will be in the output; any test case that would normally have that one value will have one of its names instead. Names are rotated among the test cases.

Allowing bundling sub-models into groups

The default behavior for PICT is to generate N-Wise combinations for all system inputs defined in the input file using the "***/o:* N"** argument. However, PICT offers the tester the ability to identify certain sub-set of inputs to have different N-Wise combinations than the rest of the inputs. For example, the tester may need to generate 4-Wise combination for Parameter 1,2,4, and 16, while having the rest of inputs 3,5-to-15 as pair wise. To do so, add the following line to your system input.

{<ParamName1>, <ParamName2>, <ParamName3>, ... } @ <Order>

{P1, P2, P4, P16} @ 4
{P3, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15} @ 2

Negative Testing

Allows the tester to test invalid system inputs' values; which is commonly referred-to by "Negative Testing". Negative testing usually validates defensive architectures and requires a special selection of invalid system inputs. It is highly recommended, not to combine two invalid system values in the same test case to avoid what is called "Error-Masking".

When two invalid system inputs cause the system under testing to fail, you don't have a simple way to know which invalid input caused the system to fail. In other words, one error is masking another error. PICT is smart enough, not to combine two negative system inputs in the same test case, provided that you mark certain input values as in-valid. The following line defines certain input value (-1) as invalid input to a certain system parameter (A and B).

A: ~-1, 0, 1, 2

B: ~-1, 0, 1, 2

You mark the value with "~" to instruct PICT to treat it as invalid input.

Value Weighting

Using weights, a tester can instruct PICT to prefer certain values. A weight can be any positive integer and the greater the number the more chance the PICT will prefer it, provided that N-Wise criteria are satisfied. See the following example, where we prefer the marital status to be "Single" provided that the rest of values appeared previously in test cases in a way that satisfy N-Wise criteria.

Marital Status: Single ***(10)****, Married, Divorced,*

Conclusion

PICT is a very easy but extremely useful testing tool. It is being used for testing systems regardless of the system size. PICT is a batch tool, which can be easily integrated in way that automates the test case design since every thing is an argument to PICT. For more details, visit Microsoft Research site and download the tool: (http://download.microsoft.com/download/f/5/5/f55484df-8494-48fa-8dbd-8c6f76cc014b/pict33.msi)

References

[1] http://www.pairwise.org/

Biography

Omar Kamal, 12 years of experience in wireless telecommunications, software development, training and software quality management. He holds a bachelor's degree in telecommunications engineering from Cairo University, and master's degree in business administration from City University. In addition, he is a "Certified Quality Manager" by the American Society for Quality.

Feedback Contacts

Feedback, comments and questions are appreciated by the author.

Email:
omkamal@yahoo.com

The Importance of Sound Software Engineering Practices Personal Software Process (PSP)

By: Mostafa Hamza

"Reliable and transparent programs are usually not in the interest of the designer." *Niklaus Wirth, a Swiss computer scientist and the designer of Pascal language.*

Actually, this is the problem with the software industry. Due to the lack of software process or engineering methodology, the produced software is not of good quality that does not last long. To achieve the software ultimate goal, there is a need for a well-defined process or engineering methodology that we follow. Software engineering in its meaning is to provide good and well-structured software with high quality to be easily maintained, updated and be reusable. One of the main targets of software engineering is to minimize the cost.

To achieve such goal, there is a tradeoff between the cost and the quality of the produced software. Quality of the software pushes its cost to exceed the specified budget. Therefore, there is a need for software engineered process to follow for alleviating the risks and issues that could occur during the development of produced software.

By applying the software process, in my opinion, the produced projects will be delivered with better quality and in better timing rather than the ones that are without process. Planning, Designing, development and testing phases are the core of any software process as stated by the Personal Software Process (PSP). These phases could be extended and customized according to every company and the work environment. The following sections will give a brief overview of each PSP phases.

Planning phase:

This phase is for covering all the aspects of the project and estimating the time for each and every module in the project. It should contain time for conceptual or preliminary design that is done by the team for each module without getting much deep in the design.

This phase helps in defining and estimating the time for a project to take from the beginning without estimating wrong resources and time for the project. Developing test cases in this phase helps in knowing all the problems and the flaws that could escape without capturing them in the design. Also, this could improve the security and quality of the produced software.

Designing phase:

Designing phase is for producing clear and precise design that the development phase should follow. The design should verify the conceptual design done in the previous phase. If there are defects found here, the cost of fixing it will be much less than if it escaped to a later phase.

Therefore, this phase should take enough time without getting into the development. The variability in Software processes helps the engineers to choose the best suitable

process to follow. Some of the processes are not efficient in certain context or project while others could be effective. In my opinion, I think the iterative model could be the best because it helps in dividing the software into modules. Modules then are divided over the iterations according to the importance. Teams could attack the bigger and more risky modules at the beginning to make sure they solve any problem that could happen.

UML is very helpful in this phase. There are many tools that create UML diagrams and could convert this design into code, but unfortunately without getting in the business logic.

Development phase:

Development phase is the phase for the coding. If the design got into all the details, this phase should not take much time and it should go smooth and easy.

Testing phase:

Testing phase is for having extensive testing that guarantee that there are no defects in the final release of the project. The more the defect stay in the system the more time it will take to recover from. Also, unintentionally the team could inject more defects while trying to fix this one. The produced test cases that are developed in the planning phase should be run in the testing phase.

Other habits should people take care of while developing such as documentation. Documentation should be a part in the process, and its time should be added to the project time. The documented code will help in the case of maintainability or readability. The software industry is always on change, people join the project and others leave.
If there are no documents for the work done, there will be great mess. Such documents should be as a reference to the project members. The newly added resources to the project should also review such documents before joining to help in understanding the written code rather than starting from scratch or injecting bugs unintentionally.

Packaging the produced code that is developed in-house into libraries is another habit that teams should take care of. This will increase the reusability rather than starting from scratch. For example, in a software house that develops web-based systems. Mostly of the done work is the same across the majority of the projects running. The differences could be so limited. Developing abstract layout or framework for the similarities with the option of customization will definitely decrease the development time.

References

[1] http://www.pairwise.org/

Biography

Mostafa Hamza, is a junior software developer in LINK Development. He holds a BSc. In computer science from the American University in Cairo (AUC) and is an SEI-Certified PSP Engineer.

Feedback Contacts

Feedback, comments and questions are appreciated by the author.

Email:
moustafa.hamza@mail.link.net