Support Vector Machine based, project simulation with focus on Security in software development Introducing Safe Software Development Life Cycle (SSDLC) model

Preeti Mulay
Bharati Vidyapeeth University
Pune

Dr. Parag Kulkarni
Capsilon
Pune

Summary
Our proposed methodology introduces new concepts in the areas of security. This methodology also focuses on classification using SVM principles, and estimating complete details of SSDLC. This way the complete team will get to know in advance, before even project begins, complete simulation. As every phase of SSDLC is handling security aspects, software application developed will be more efficient and effective. This way of software development will reduce dependency on security team. Following proposed SSDLC model will empower software development team and increase their confidence levels; thereby decreasing stress and hence better timely output. Proposed chain structure of passphrase is another more suitable authentication technique as compared to one passphrase or use of password. Use of suggested "Hinglish" is also the best suitable practice to follow, at least in country like India. Other countries also may follow similar concept. Use of such combination of languages will be difficult for hacker to hack. Introduction of "Safe Cases" is one more positive way of looking at things. Developing "Safe Cases" will require expert to understand clients complete detail network, topology, systems etc. Based on this information and updated details about hacking, experts should develop cases to secure client's application, data and network.

Keywords: Security, SVM, SDLC, cluster, simulate, life cycle, software development

Abstract
Safety or Security is one of the important requirements of Software development industry. To implement safety at development level to produce a trustworthy application, proposed Safe Software Development Life Cycle (SSDLC) model is important. This is feasible by implementing safety at every phase of SSDLC by developers themselves, without waiting for security team to implement / insert required items, later. In our research work, after collecting the complete (new software) project information, it is feasible to apply Support Vector Machine (SVM) principles to classify given / available data and then simulate new project in front of the whole team. One of the ways to classify using SVM is given in coming sections. Our research include making use of SVM's principles to classify available data, special focus is on Security aspects, simulation of new project details (estimation), forming new clusters if required, reuse of available classes, objects, documents, use cases etc. This classification of data and information can help us in identifying the points of vulnerability. This paper also suggests better ways to achieve the security in software development based on pattern classification. This security includes system security, application security and network security.

Related work
On the applabs's website, authors mentioned that "There is a world of difference between feeling secure and being secure", which is a reality in software application development. As mentioned in his book [5] Software Security, by Gary McGraw, many aspects of security including network security, system security, application security etc are all important aspects. We also need to understand Privacy of data, organization and individuals as another feather attached to security, as given in [6] IEEE Security and Privacy. It has been estimated that it is about 200 times more expensive to fix a problem when an IT system is in Production compared to fixing at the requirements analysis [1] step during Development. The factor falls to about 4 for small IT projects but can exceed 500 for very large projects. Even if these figures are only vaguely close to the truth [9], the implications for quality assurance processes in IT development are crystal clear, as are the benefits of splitting massive projects into discrete sub-projects. [7].According to a Gartner Report, 75% of attacks today occur at the application level. A Forrester survey states that "people are now attacking through applications, because it's easier than through the network layer." To identify, analyze and report vulnerabilities in a given application, comprehensive risk assessment solution is must. [12]. I feel every client who wishes to get software application developed from an organization need to have clear picture about all security concerns. The client should be able to provide security details to development team. Once these details are available (referring to new software project), simulating complete project by following classification and forecasting (based on available historical data) will be easy. That's the goal of this paper.

Manuscript received November 5, 2008
Manuscript revised November 20, 2008

1. Introduction

My research work specially focuses on implementing principles of SVM to develop software project more efficiently and effectively. One more utmost important aspect is security implementation (suggestions) at every level of SDLC. If the software development team is able to understand whole requirements of new software project; handling project development work will become bit easier. Hence I would like to propose a new tool which will give details of new project by learning from history of that organization and team. Every organization has their own network setup, firewall, servers and antivirus setup ready. My tool is capable enough to handle security at every level, even before classification begins. Here is the summary of included sections in this paper. Section 2 describes proposed SSDLC model, section 3 defines phase-wise security issues, section 4 explains about SVM principles, section 5 talks about my suggestions and followed by conclusion. In addition to this there is one more feature involved in our research. Based on available historical data [2][3], formed clusters, and new project's feature vector, we can implement behavioral patterns. Once these patterns are ready we can learn from these patterns, study these patterns to estimate project details. Once empirical data is ready, we will automatically get representative data series and pattern. When new feature vector arrives, tool starts comparing with this representative data series and pattern. We can learn from just looking at that pattern.

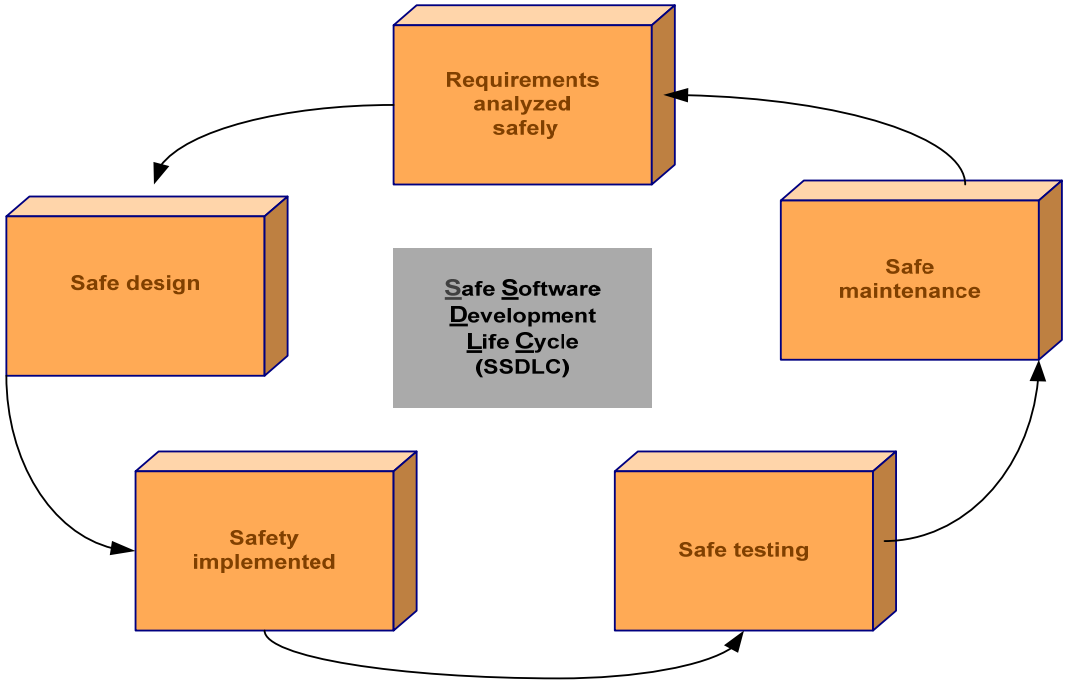


Figure 1 shows new proposed *S*afe *S*oftware *D*evelopment *L*ife *C*ycle model.

2. New proposed SSDLC Model

I feel the time has arrived to make changes in SDLC model. Now security aspects need to be incorporated in SDLC models in a very safe manner and hence suggesting this model which is called Safe SDLC. This model includes analyzing requirements safely, developing safe design; implement every module following security aspects, developing "safe cases" in addition to "Test cases", "use cases" and (instead of) "Abuse cases". Even the maintenance of any code here onwards needs to take care of safety and security. While implementing all these aspects developer need to definitely focus on functional and non-functional requirements given by client, and quality as well. During safe maintenance phase, if required, developer will go back to requirements analysis phase and continue making changes in other parts of SSDLC also. If software application is developed by using this model, then dependency on security team will be reduced, time will be utilized better, developers will have more satisfaction and confidence about their work, and hence finally organization will be able to produce more effective and efficient application quickly than ever before.
To implement this at every development stage, let's learn some of the important related and useful definitions.

3. Security related definitions(phase wise)

This section gives some important definitions [1] related to security at various phases of SDLC model. These phases include requirements, design, testing, implement, and maintenance.

Abuse cases at Requirements phase

Principally developers develop use cases, and the concept of abuse cases is derived from these use cases. After implementing use cases, next step is to represent these details pictorially to increase understandability; via use case diagrams. While developing abuse case, expert needs to assume to misuse software application and handle its corresponding effects. Expert need to think from hackers' perspective and find out loop holes, if any, in an application environment.
Every abuse case depicts such cases and as well information about how to handle them. These details need to be reflected (might be a part of it) in test cases also.

Safe Cases (all phases)

How ready our software application is to tackle all possible hacking issues, is major focus of "safe cases"; looking into positive way of dealing with things. How to prevent misuse of developed application at all levels, including data, network, database, distributed systems, users, sharing, messaging etc. should be the major concern behind developing "safe cases". While developing / designing safe cases, experts need to think from clients (its environment / platform) point of view.

Business risk analysis at Design phase

Most important part of any SSDLC model is to analyze risk. Find out what type of risk is expected. Here comes the important use of my research work. Finding out what type of risk to expect is feasible as organization need to maintain empirical / historical data, my tool classify them, store them for future reuse etc. This tool is also capable enough to form a separate risk based class. It is feasible to caution complete team if any risk is likely to be associated with new project, once feature vector is made available.
A good risk analysis considers questions of the project's cost to the parent organization sponsoring the software in terms of both direct cost (liability, lost productivity, and rework) and indirect cost (reputation and brand damage).

Architectural risk analysis at Design phase (HLD, LLD)

Similar to a business risk analysis, an architectural risk analysis assesses the technical security exposures in an application's proposed design and links them to business impact. Starting with a high-level depiction of the design, the analysis team considers each module, interface, interaction, and so forth against known attack methodologies and their likelihood of success. To provide a forest-level view of a software system's security posture, the analysts typically apply such analyses against a design's individual subcomponents as well as to the design as a whole. Attention to security's holistic aspects is paramount: at least 50 percent of all security defects are architectural in nature.[1][6].

Security functionality testing at Testing phase

Just as testers typically use functional specifications and requirements to create test scenarios and test plans (especially those testers who understand the critical notion of requirements traceability), security-specific functionality should be used to derive tests against the target software's security functions. These kinds of investigations generally include tests that verify security features such as encryption, user identification, logging, confidentiality, authentication (with chain of passphrases / passwords, can be in combination of various languages), and so on. These are "positive" security features for white hats.

Risk-driven testing at Testing phase

Risk-based test scenarios are the natural result of the process of assessing and prioritizing software's

architectural risks. Each architectural risk and safe case considered should be described and documented down to a level that clearly explains how an attacker might go about exploiting a weakness and compromising the software. Such documented results can be reused later, in future. Our tool provides detail link of all reusable items, after comparing developed clusters with feature vector. Such descriptions can help generate a priority-based list of test scenarios for later "adversarial" testing. Experts need to think and develop both successful and unsuccessful test data.

Code review at Implementation phase

The design-centric activities described thus far focus on architectural flaws built into software design, but they completely overlook implementation bugs that the coders might introduce during coding. Implementation bugs are both numerous and common and can include nasty creatures such as the notorious buffer overflow, which owes its existence to the use (or misuse) of vulnerable APIs. Code review processes—both manual and (even more important) automated with a static analysis tool—attempt to identify security bugs prior to the software's release.

Penetration testing at System testing module

System penetration testing, when used appropriately, focuses on human and procedural failures made during the software's configuration and deployment. The best kinds of penetration testing are driven by previously identified risks (maintained details in the form of cluster, using our algorithm and tool) and are engineered to probe risks directly to ascertain their exploitability.

Deployment and operations at Field system module

Careful configuration and customization of any software application's deployment environment can greatly enhance its security posture. Designing a smartly tailored deployment environment for a program requires following a process that starts at the network-component level, proceeds through the operating system, and ends with the application's own security configuration and setup.

Threat model

Developers should also define their application's *threat model*, which describes the possible threats that can occur in a given security environment. One of the most common ways of finding threats is to use the Stride categories (for *s*poofing, *t*ampering, *r*epudiation, *i*nformation disclosure, *d*enial of service, and *e*levation of privilege) or one of the 49 attack patterns identified elsewhere. We can then build an attack tree to detail each threat, with the attack's goal represented as the tree's root, and leaf nodes representing different ways to achieve that goal.

Coding error

Recent statistics show that programmers write one bug per 55 lines of code, and as with any other piece of code, cryptographic implementations are likely to contain bug.

Misunderstanding or misuse of the algorithm

In some cases, the algorithm is fine, but it's being used for something it wasn't meant to handle.

Now as all the definitions are known and what is required to handle security is learnt, let's concentrate on classification using SVM principles.

4. SVM principles and classification

To understand SVM principles regarding patterns and classification, let's first take walkthrough of an SVM quantitative explanation.

An *n*-dimensional pattern (object) x has *n* coordinates, x=(x1, x2, …, xn), where each xi is a real number, xi∈R for i = 1, 2, …, n. Each pattern xj belongs to a class yj∈{-1, +1}. Consider a training set *T* of *m* patterns together with their classes, $T=\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$. Consider a dot product space *S*, in which the patterns x are embedded, x1, x2, …, xm∈S. Any hyperplane in the space S can be written as

$$\{\mathbf{x} \in S \mid \mathbf{w} \cdot \mathbf{x} + b = 0\}, \quad \mathbf{w} \in S, b \in R$$

The dot product w•x is defined by:

$$\mathbf{w} \cdot \mathbf{x} = \sum_{i=1}^{n} w_i x_i$$

A training set of patterns is linearly separable if there exists at least one linear classifier defined by the pair (w, *b*) which correctly classifies all training patterns (Figure 2). This linear classifier is represented by the hyperplane H (w•x+*b*=0) and defines a region for class +1 patterns (w•x+*b*>0) and another region for class -1 patterns (w•x+*b*<0).

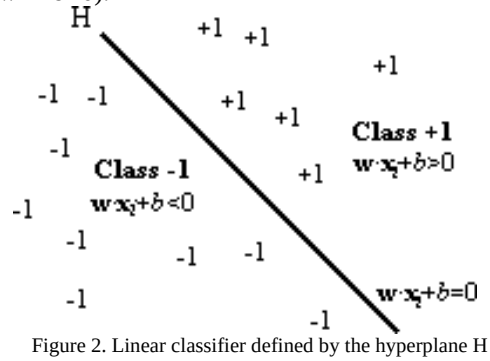


Figure 2. Linear classifier defined by the hyperplane H (w•x+*b*=0).

After training, the classifier is ready to predict the class membership for new patterns, different from those used in training. The class of a pattern x_k is determined with the equation:

$$\text{class}(\mathbf{x}_k) = \begin{cases} +1 & \text{if} \quad \mathbf{w} \cdot \mathbf{x}_k + b > 0 \\ -1 & \text{if} \quad \mathbf{w} \cdot \mathbf{x}_k + b < 0 \end{cases}$$

Therefore, the classification of new patterns depends only on the sign of the expression w•x+b.
Hence if the features required for new project is already available and handled by developing organization (during previous projects) then this shows + sign of expression, otherwise we need to formulate a new cluster. Repeat these steps with every new project and this is how I believe we can classify new project's data or features using SVM's principles.
Section 3 and 4 illustrated security aspects and classification principles. Section 5 gives details about set wise proposed algorithm to achieve our research aim.

5. Proposed methodology

I would like to implement "safe cases" instead of "abuse cases". It will be similar to "abuse cases" but will have positive ways to handle attacks.
As explained in [13] by authors about concept called "Passphrase", I would like to make few suggestions. I will explain these by taking one example also. In [13] author said **PASSPHRASE need to be used instead of passwords.** They also said that **Passphrase is more secure than a password because of its length.**
The better way is to use chain of GUI's to accept **Chain of passphrases / passwords after first / initial verified passphrase.** I sincerely feel that using A Passphrase will not solve security issues, but if organization decides to implement chain of passphrases, it will become difficult for hackers to break this security aspect.
In his book on security issues in [1] and [5], authors did also mention about use of "native language" for developing "passphrase". Here is a paragraph from the book:
"Using your native language is probably an obvious choice. Throughout this FAQ, data and statistics apply to English text. Using another language or combining languages will change the numbers some. It will not make your passphrase harder to guess. Attacking a different language or even multiple languages is still the same. The search space is roughly the size of the language or grows by adding the size of the average size of the vocabulary of the added language. Dictionary attacks in another language would run in the same manner as a dictionary attack in English. "

I don't agree to this explanation given on this website. Let me tell you what happens now days in India. Our national language is "Hindi" and we use English also. In India every state almost uses their own language. The young generation even while typing even SMS uses their own native language but type those words using English. For ex. I am using a word in Hindi, but typing it using English alphabet, so it becomes Hinglish. In such cases neither English nor Hindi dictionary is useful for attackers. As these words will not be available in any of the dictionaries, unless there is one available in Hinglish, and the chances are very rare.

After suggesting new SSDLC model, use of native languages for passphrase, "safe cases", let's see the steps involved in our algorithm.
This tool is GUI based, can be kept on organization's intranet website or internet website. The steps of algorithm are as follows: organization using this tool is ready to know the features of new software development project.

- Enter information received from client, begin with PROBLEM DEFINITION
- Check, verify and validate for security related information including standards to follow
- IF MISSING, then get those details first, may need to contact client or check available historical database
- Once entered, then check the feasibility of required security standards
- Then start with REUSABLE CLUSTERS, and formation of NEW CLUSTERS, if required
- Estimate and simulate project details
- Suggested security requirements, if some of the security aspect is missing, based on available historical data.

Figure 3 below shows the pictorial representation and description of this algorithm.
We will make use of nearest neighbor, k-means or our own algorithm called "closeness factor" as a quantitative method. [2].
As this tool is available on organizations website or intranet site, hacking is very much possible. I would like to explain by taking few examples and scenarios.

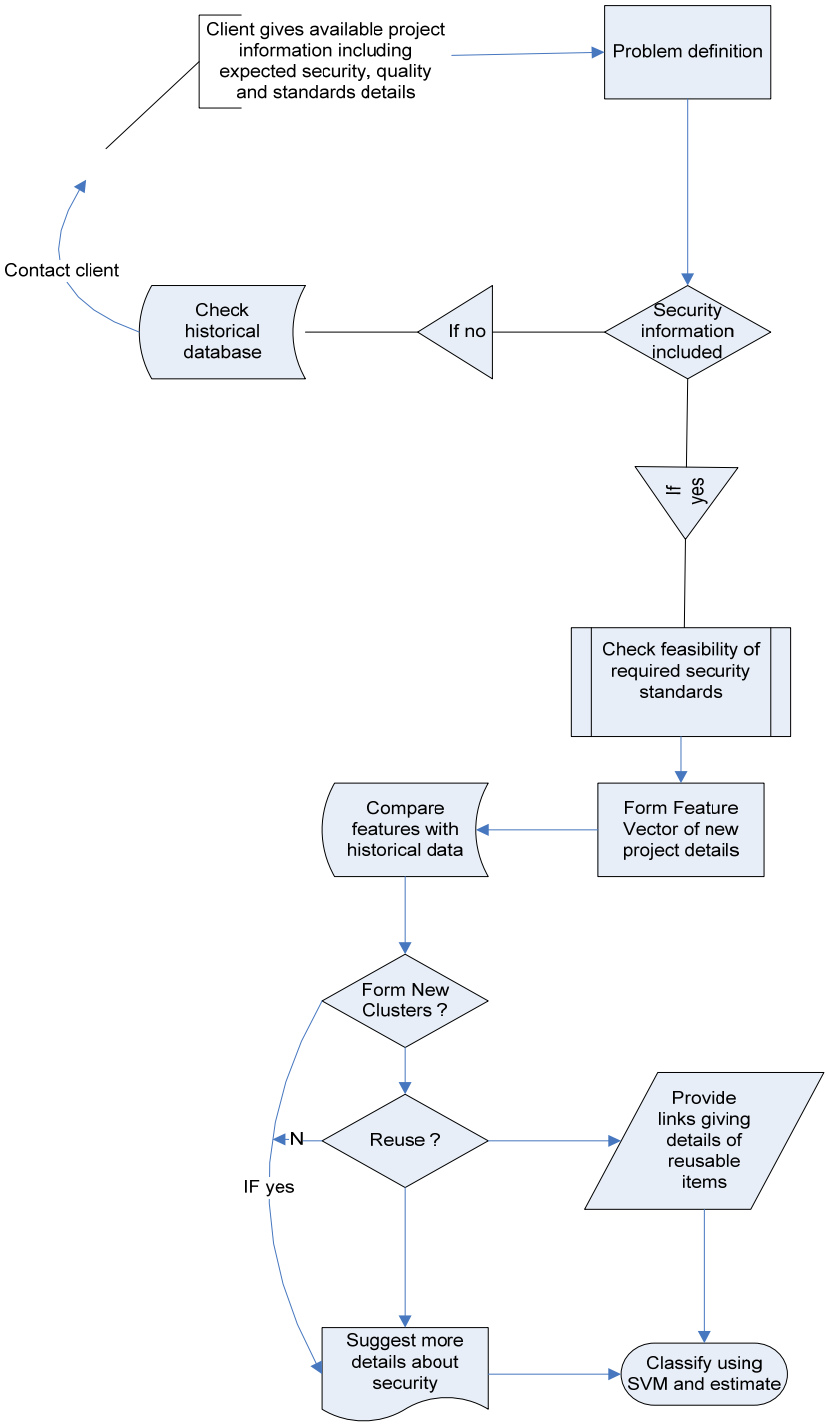
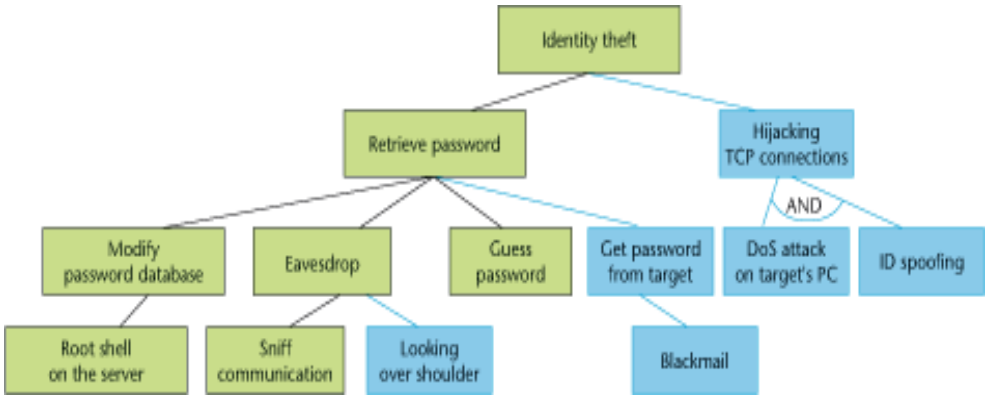


Figure 3 shows proposed methodology details, given in algorithm.

Assume that complete team is using this tool available on organizations website, to learn more about new project and problem arises. Hacker hack the website and specially dealt with problem definition given by client [this is just an example scenario]. The new project's client belongs to banking domain and uses 10 different products. Software development organization is expected to develop this code fulfilling 10 products, but hacker hacked the details at problem definition level itself and deleted few contents. Before the development team could able to read and implement further details, only information available in front of them was: client deals with 5 products only. Now what. Unless and until organization verifies from client at every step, the team will concentrate on developing only 5 products and hence "Safe cases" to handle security should be available at every stage.

To further continue with this example / illustration if team develops assuming only 5 products, clustering, classification, testing, reuse, risk analysis, forecasting everything will happen assuming only these 5 products. Whole set of efforts may go waste and hence security assurance
from the first initial phase of product development, understanding clients networks, dealing with
clients distributed database management system (if any, never would like to do DBA's job),
distributed operating systems concepts, need to be considered. In addition to forest view, developing "Safe cases", penetration testing is an important criterion to follow.

Another example is based on one of the leading internet service providers of USA, "America Online". As discussed in [1][4][6], America online team identifies theft in following manner as represented in figure 4. Theft identification is feasible either by retrieving password or hijacking TCP connection. There are various ways of Retrieving password, which include (as given in the diagram, guess password, modify password database, get password from target etc.



America online's Figure 4. Threat model for identity theft. The boxes with blue lines are eliminated at the risk evaluation step.

America online have defined two main security objectives: integrity protection of exchanged messages and client authentication. I feel client authentication should be by using linked pass phrases or passwords instead of simply passphrase / passwords, as suggested in our proposed methodology.

After discussing all these major security concern related to our research in particular and software development in general, let's discuss advantages of our proposed methodology.

References:

[1]Bridging the Gap between Software Development and Information Security Kenneth R. van Wyk and Gary McGraw, Cigital and KRvW Associates, Cigital

[2]M. Kulkarni, P. Kulkarni, Advanced forecasting methods for resource management and medical decision-making, National Confernece on Management Future trends, Hyderabad, India, February 2002.

[3] P. Mulay, P. Kulkarni, "An Automated Forecasting Tool (AFT) achieved Clustering Entity Relationship Model". International Journal for Computer Science and Network Security (IJCSNS)- Jan 2009

[4] http://www.noticebored.com/html

[5]Software Security, By Gary McGraw

[6] Misuse and Abuse Cases: Getting Past the Positive, IEEE Security and Privacy
Volume 2 , Issue 3 (May 2004), Pages: 90 – 92, Year of Publication: 2004

[7] G. Sindre and A.L. Opdahl, "Eliciting Security Requirements by Misuse Cases," Proc. 37th Int'l Conf. Technology of Object-Oriented Languages and Systems (TOOLS-37 '00), IEEE Press, 2000, pp. 120–131.

[8] I.Alexander, "misuse cases : Use Cases with Hostile Intent", IEEE Software, Vol 20, no. 1, 2003

[9] J.Viega and G.McGraw, "Building Secure Software, How to Avoid Security Problems the right Way",Addison-Wesley 2001

[10] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323, 1999.

[11] Chris Ding, Xiaofeng He, Hongyuan Zha, and Horst Simon. Adaptive dimension reduction for clustering high dimensional data. In *Proceedings of the 2nd IEEE International Conference on Data Mining*, 2002.

[12] http://www.securityfocus.com/infocus/1783

[13] http://www.iusmentis.com/security/passphrasefaq/practical

An alumnus of IIT and IIM, Dr. Parag completed his Ph.D. in Computer Engineering from IIT Kharagpur. He has been working in IT industry for last 17 years. He has worked as Research head, operations head, GM, Director and was instrumental in building world-class software product companies.. He is working as a Vice-President Strategic Development and Chief Scientist at Capsilon INDIA. His name and profile is selected for listing in "Marquis Who's Who in the world" (Science and Engineering) –2009. He has written many business articles. He has more than 60 International publications and two patents pending in US PTO. He is member of IASTED technical committee, WSEAS working committee, board of studies of two institutes and is guiding 7 Ph.D. students. Parag has conducted more than 25 tutorials on research and business topics at various international conferences He is visiting faulty at IIM Indore. He is pioneer of new management program "Deliverance from Success" for Executives and author of books "Deliverance from Success" and "IT strategy". His areas of research and product development include M-maps, intelligent systems, text mining, image processing, Decision systems, Semi-constrained influence diagrams, forecasting, quantitative analysis, knowledge management, IT strategy, classification, distributed computing, AI and machine learning.

Prof. Preeti Mulay is working on her PhD in the areas of "software engineering". She completed her MS (Software Engineering) from Wayne State University, MI, USA 2002 and M.Tech (Software Engineering) from JNTU, India, 2000. She is working in the education field since 1995, on various positions. Her areas of research include Software Engineering, pattern matching, forecasting, knowledge management, clustering, and machine learning.