International Journal of Electronics Communication and Computer Engineering
Volume 6, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Model Driven Software Engineering in the Mobile Era with an Emphasis on Security

M. Bhanu Sridhar

***Abstract* – Software Engineering is a prominent field of Computer Science that can be used to organise and implement a software project with planned tactics. Handling of security concerns in all phases of Software Development Life Cycle (SDLC) is the critical need of the hour. Mobile Computing allows the transmission of data through any wireless device and is overlapping all the fields in Computer Science. This paper makes a detailed study of SDLC models in the current Mobile Computing era, with more emphasis given on security so as to compose the models more applicable in the industry. A new Software Engineering Model (SEMO) or Z-Model is proposed, that takes care of the security aspect.**

***Keywords* – Software Engineering, Models, Mobile, Security, SDLC, Testing, SEMO.**

I. Introduction

Software Engineering took shape in 1960s [1] when no proper planning existed for the development of software projects and applications. Over the years, many models and methodologies evolved [2] and established themselves as benchmarks in the software development field. Popular models like waterfall model, V-model, spiral model, rapid prototyping model, object oriented model and the agile process have taken their places in the industry and were utilized by the developers as per their requirement. These models basically guide the Software Development Life Cycle (SDLC) through a planned process, through which more positive results can be expected. The different fields of SDLC like requirements, analysis, design, coding, QA (testing) and maintenance are organized in a precise way by the models to reach their respective goals.

Though the models have done their job in an expected manner, lately the issue of security is engulfing the IT industry. The SDLC models indicate a planned path of development of a project but it might be observed carefully that no particular emphasis is laid upon the hotspot of IT these days – security. At the each phase of a model the developer or user are given a path with nothing specified about security. Methods that talk about security do exist but they consider security aspect at only the starting phase [3]. It's time now to revise the existing SDLC models to indicate the security point at any phase or at all phases so as to bring out a full-fledged 'secure' model into the field.

Another point to be noted these days is that everything concerned with the IT industry is getting into the palm of the users as apps on mobile devices. Different fields like e-commerce, banking, data analysis etc. have embraced the mobile field so as to make them more available to the user. This increases the need of a secure model for the development of the applications, laying more stress on security.

In this paper, the existing SDLC models are studied briefly so as to pinpoint the pros and cons and to highlight the absence of the security aspect. After the study, a 'secure' waterfall model (SEMO) taking care of the security part is proposed. The paper is organised as follows: Section-I of the paper deals with the introduction; four existing SDLC models are studied in Section-II; in Section-III, the security aspect in SDLC modelling is presented as the need of the hour, particularly for the mobile app development; Section-IV presents the new modified waterfall model with security firmly taking its place and finally the Section-V presents the conclusion and future enhancements.

II. Study of Models

The Software Development Life Cycle is a process utilized for planning, creating, testing and deploying an information system [4]. The basic idea in the SDLC is to devise a well organised methodology for a software developer. It takes into consideration the requirements of the user, analyses and designs the development process, proceeds with coding and testing and finally the installation and maintenance parts of the concerned software project are also taken care of. To supervise this level of intricacy, different models for the process have been proposed and utilized. In this paper four models are studied briefly, highlighting the absence of the security aspect at every phase.

A. Waterfall Model

The Waterfall Model is a well known approach to the implementation of SDLC and was proposed in 1970 [5]. This model takes care of planning in early stages and proceeds from a phase to another after its completion. The classic waterfall model had the flaw that if anything had gone wrong at early phases, it could only be discovered at the final phase. To remove this predicament, many updates to the model were proposed and one of them can be seen in Fig.1. Different modifications have provided to different options to the developer to come back to the previous phase or to the starting phase.

This model starts at the requirements phases that are acquired from the user. The system and software requirements are then deduced form the real needs of the user and are refined until the project team and the user are satisfied. The analysis phase deduces in a planned methodology of what exactly is to be done and the design phase infers how the analysis is to be carried out.

Testing is a critical part of the model where each program is rigorously tested, errors are unearthed and the suggestions are sent back to the coding team. The project is finally approved by the testing team if not much errors/bugs could be found.

International Journal of Electronics Communication and Computer Engineering
Volume 6, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

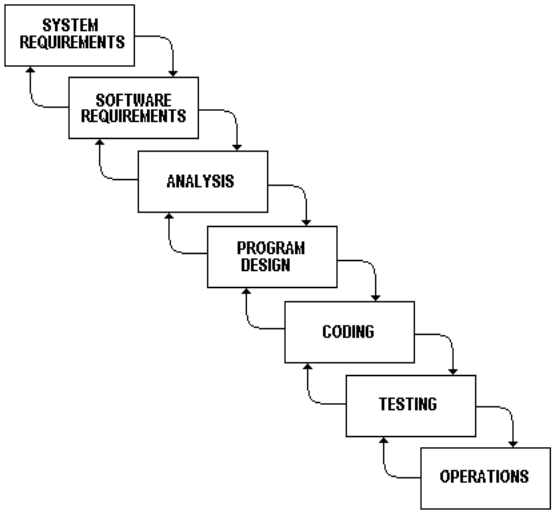

Fig.1. Modified Waterfall Model

After all these background operations, the project is ready in hand and then the final operations like installation, maintenance and retirement will be done.

At each phase, documents are created and filed, which explain the requirements for that phase. At the end of a phase a review is conducted to decide whether to proceed to the next phase or not.

A pure waterfall model is not applicable in all cases since no overlaps between the phases could be allowed. Modified waterfall models have taken care of this flaw and proceeding to different phases is possible now. Still, the model is noted to be an idealized one, not matching the reality well. Also the requirements at different phases may change from time to time, causing chaos in the SDLC process.

It may be noted that no mention of security could be found neither in classical nor in modified waterfall model(s). Security has to be given the utmost importance since almost every day we learn about many websites being compromised due to hacking. Simple mention of security testing doesn't take care of this aspect and inducing security into the SDLC as an integral part is the need of the hour.

B. V-Model

The V-model can be considered as an extension to the waterfall model. In this case, instead of moving down linearly, processing pursues a V shaped path, moving up after the coding phase.

The V-Model contains two phases, namely Verification and Validation. Each of the aspect in these two phases is inter-connected where at the end of an aspect in Verification can be tested in Validation. It should be noted that Coding forms the central part in this model.

This model exhibits the relationships between each phase of the SDLC and its own type of testing [6]. Testing, at different levels, is associated with all the phases and apparently, provides good Quality Assurance. The model itself can be considered to be exhibiting two phases: verification and validation. In verification, requirements are analysed, system design is proposed, architecture and module designs are also finalised with appropriate testing. In validation, different tests are carried at different levels i.e. unit testing, integration testing, system testing and acceptance testing to be assured of the proper transaction of the project.

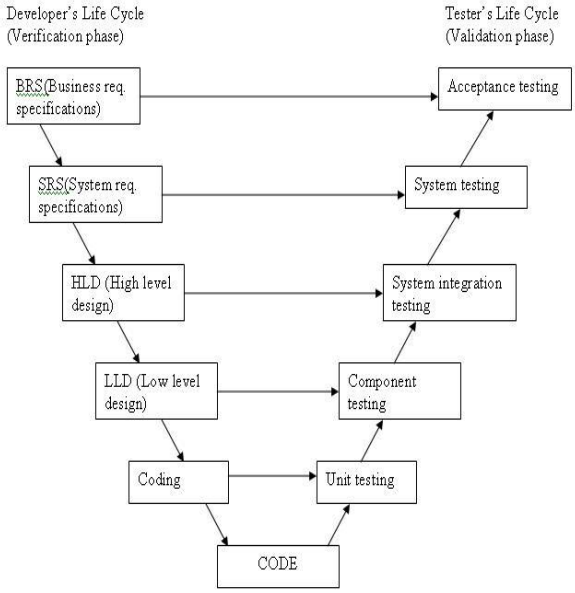

Fig.2. V-Model

Still, this model has also been criticised by the experts. It is said that this model gives a false sense of security, doesn't respond to changes, testing scripts of a phase are written well in advance before the project reaches the concerned phase and so on.

It should be noted that the testing done at different levels only brings out the errors, bugs or whatsoever but doesn't stand rock-solid to attacks online by the black hats. In the present mobile era there is quite a good chance that any app can compromise any user for any type of data.

C. Spiral Model

The Spiral Model was proposed by Barry Boehm [7] in 1986 and it creates a risk-driven approach to the software process rather than document-driven or code-driven process.

At each level of the project from requirement to implementation, risk analysis is conducted to obtain a prototype as the final output. The prototype is tested and modified till minimum bugs are to be found. The final prototype at the end of the model will be the main project itself. Note that incremental, waterfall and other process models can be considered as special cases of the spiral model. Starting at the baseline of requirements phase, risk analysis is applied at the end of each phase, in which a process is undertaken to identify risk, and provide the alternative solutions [8] with the concerned prototype.

Though the security aspect might be considered to be embedded in the risk analysis, the user needs to be an expert in the concerned phase. The model is also highlighted as being time-taking and costly. For the smaller projects these days like developing mobile apps, this doesn't fit the bill.

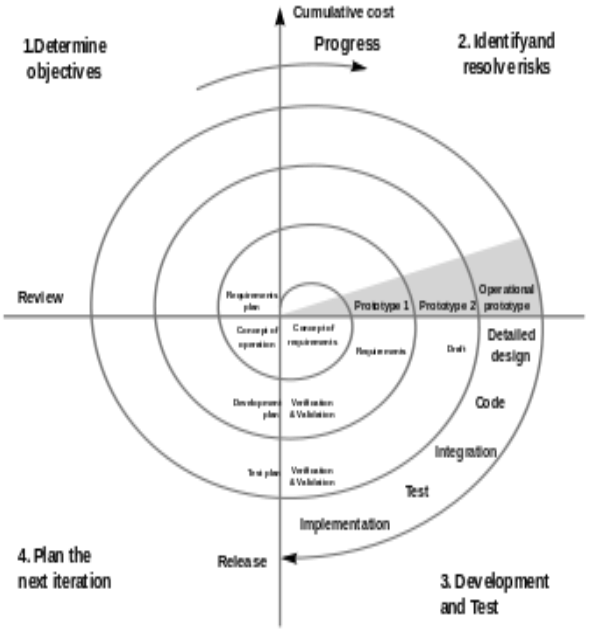

Fig.3. Spiral Model

D. Agile Software Development

Agile software development represents a major departure from traditional, plan-based approaches to software engineering [9] and depends more on the developers' creativity rather than on processes [10]. In this process, the concerned team interacts with each other (or as groups) over the processes and tools that are to be used. Still it might be noticed that comprehensive documentation is produced here too for the software usage or development. But the important new feature here is constant customer collaboration and responding in time for changes in the project plans.

A limitation that is being mentioned lately in agile process is the less interaction between different groups where members are not much interchangeable. Still, agile process gives enhanced code quality, positive job satisfaction and good response to any unexpected changes in the plans.

As a compare, waterfall model is *document-driven* and V-Model is *code-driven*. It has also been stated that spiral model is *risk-driven* and agile method is '*discussion-driven'*. In all these models, the IT field is now abuzz with the agile model these days. But again we can note that even this agile model gives only limited output as far as future security of the project is concerned. This idea gave rise to the SEMO in the *current mobile period* where dangerous (security) bugs arise from the cases of hacking, virus or pointed out by different labs like Kaspersky, Google Zero etc. This paves the path for the birth of Security Model (SEMO).

III. MOBILE SECURITY ASPECT

A. Security in SDLC

Though the security facet hasn't been given more light in the SDLC models, recently the trend appears to be changing. The attacks that are being launched on different software applications after their deployment, generally online, are hugely extensive. Much study was conducted on network security but all those were confined for discussing how to provide proper encryption for the data that is in storage or transmission. Though some studies have been conducted about encompassing the security into the SDLC models like [11] and [12], this angle in SDLC modelling needs extensive researching and implementation of the outputs as models.

The general phases of the SDLC are dealt with in a new methodology that has security at the helm. As a start when the project is at its requirements phase, the concerned team itself should not be given all the rights to access the data – this should be confined to the team leader. In the user interaction and finalization of the requirements also, this point needs to be adapted. The storage of the data or files in the cloud and its security is an entirely different story that needs its own hectic work.

The analysis and design phase where different tools, may be online tools, can be used to develop and finalise the Unified Modelling Language (UML) diagrams should also be carefully taken care of. Actually at these three starting phases of SDLC, a limited access to the data should be the most that a team-member can obtain. All the creations, modifications and deletions should be in the hands on the team leader/manager.

The important phase of coding should be apparently divided between the members and if possible, the coding team should not have any idea about the user or what exactly the project is. It is in the hands of a limited number of persons to integrate the developed modules and bring up final application. The equally important testing should also follow the same theory: proceed with unit testing and may be some integrated testing but the system testing is confined only to limited persons.

It must have been observed that the security aspect surfaces at each part of the SDLC and at even the sub-phases. Authors of [14] argue for the need to develop a methodology that considers security as an integral part of the whole system development process, but a theoretical explanation won't suffice the need of absorbing security into the SDLC models. A new model that takes of security in all the phases, particularly in mobile apps is much needed.

B. Need for Mobile Security

For any computing device that contains sensitive information and accesses the Internet, security is a major issue that should be addressed with utmost care. In contrast to the situations some 25 years back, security has taken its place firmly in the IT industry, which includes mobile devices. Keeping aside the physical security and secure data storage, the most dangerous problem here surfaces when the user gets connected to Internet for mailing, chatting, downloading apps or to access the cloud storage.

Once connected with the Internet chance of all kinds of malware, spyware and recently, the ransomware will draw their swords against the user, making him unknowingly vulnerable to the attacks. Different kinds of malware include virus, Trojan, worm, botnet and rootkits. Spyware

transfers the details of the mobile device to the hacker unknown to the user and ransomware takes the whole sensitive data into its control and demands for money to release the same.

While considering the security of a mobile computing device, it would be better to anticipate these points much before the design and manufacturing of the device or concerned software. A planned execution of a model that deals with the security aspect at all levels needs to be brought up at this juncture to reduce the vulnerability of mobile devices to the hackers.

As it was noticed in Section-2, the SDLC models that are being utilized in the IT industry previously concentrated on proper planning, cost reduce or were risk-driven. If we could induce the security feature into these models, then the models would become more accurate in the IT industry, more so in this mobile era. Instead of bracing themselves for the attacks by hackers, the mobile apps/devices should anticipate these dangers, at the planning stage itself, to counter them.

IV. SECURE MODEL – SEMO

After much consideration, the waterfall model was taken into account for improvement since it is the oldest model and would aptly serve as a launch pad for inducing security into the SDLC. The consideration of other models for this angle of improvement could come later, depending on the results that would be obtained when this new model is implemented. It could also be noticed that all the security aspects in this new model largely apply to the other computing devices also. But in this fascinating era of mobile devices, it was considered to mention the mobile devices as they form the centre point of IT industry.

The diagram of the secure model (*SEMO*) could be seen in Fig. 4. By observing the diagram, it could also be noticed as a "Z-Model'.

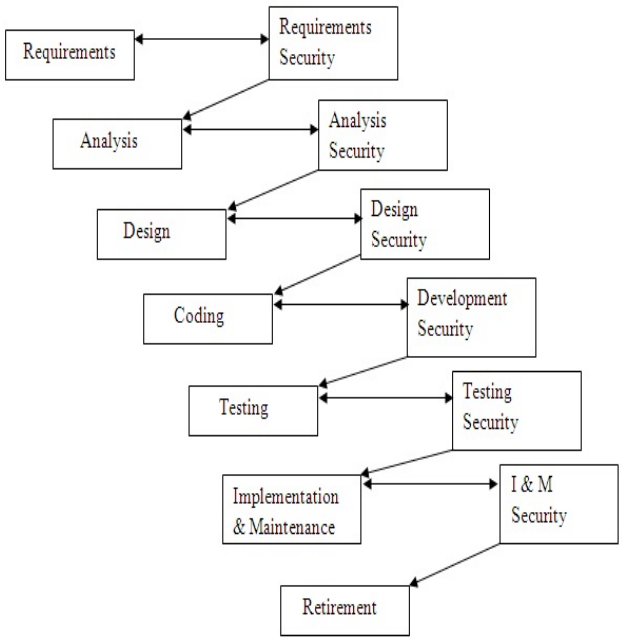

Fig.4. Secure Model (SEMO)

Through this model, it is proposed to run a *parallel security waterfall*, connected at each stage to the main SDLC waterfall. Each of the stages, including those of requirements and analysis should be designed in such a way that the security is tested before the model proceeds into the next phase. The anticipation of the security flaws that might arise in the future and devising methods to encounter them would be very handy in the mobile app industry whose growth is rocketing up every year. It is also to be noted that the baton would be passed to the next phase only by the security part of the model.

It would have been better if the security phases themselves are interconnected so as to exchange any found/anticipated problems and take the necessary steps to counter them. But in real time, this would not be as sufficient to satisfy the allocated budget. By concentrating on security, each phase from requirements through implementation is not only tested for errors but also for all possible security problems that might arise in the future.

In the IT industry, when a project is obtained, separate teams are constructed, particularly, for planning, coding and testing. It would be admirable if a 'security team', which could be a part of the testing team, is also put in place to check if the application can resist the future hacking attempts.

V. CONCLUSION

An attempt is made in this paper to bring on the security characteristic into the model driven software engineering so as to anticipate the future attacks on the application and make it ready to deal with them. Different SDLC models used by the industry were examined and it was noted that the security feature is not much to be seen in any of them. Though agility might give a chance for the team members to discuss and deduce the security measures, less interconnection between the teams might lead to a chaos.

Note that this new proposal of 'parallel waterfall' is in fact parallel security testing being conducted at all levels. The real security of a project rests on the shoulders of a 'secure' model where at each stage, the safety measures are anticipated, induced, checked and corrected to make the application rock-solid against the attacks. Even in V-Model, the testing is largely done for program errors – not security. So it can be quoted that security at each phase in SEMO can be realized as Security testing, which itself is a part of Testing.

Finally it is agreed that carrying out the security measures in parallel to each of the stage of SDLC in SEMO might result in a raise of cost and time. But the loss of data or a Distributed Denial of Service (DDoS) in a likely future attack might result not only in loss of revenue but also the reputation of the company. The security patches that are to be issued frequently for different problems would also be embarrassing. Hence the cost increase due to security measures should not be taken as purely negative.

This study aims to go much further and bring up *software security metrics* to deduce how much safe is the concerned application to any future attacks.

REFERENCES

[1] Peter Naur and Brian Randell, Software Engineering, Report on a conference sponsored by the NATO SCIENCE COMMITTEE Garmisch, Germany, 7th to 11th October 1968.
[2] Barry Boehm, A View of 20th and 21st Century Software Engineering, ICSE '06, May 20-28, 2006, Shanghai, China.
[3] Mead, N. R., E. Hough, and T. Stehney. Security Quality Requirements Engineering (SQUARE) Methodology, Software Engineering Institute, Carnegie Mellon University, 2005, Pittsburgh, PA.
[4] https://en.wikipedia.org/wiki/Systems_development_life_cycle
[5] Winston W. Royce, Managing the Development of Large Software Systems, Proceedings, IEEE WESCON, August 1970, pages 1-9.
[6] https://en.wikipedia.org/wiki/V-Model_(software_development)
[7] Boehm, Barry W. "A spiral model of software development and enhancement." *Computer* 21.5 (1988): 61-72.
[8] Munassar and Govardhan, A Comparison Between Five Models of Software Engineering, International Journal of Computer Science Issues, Vol.7, Issue 5, September 2010.
[9] Dybå, Tore, and Torgeir Dingsøyr. Empirical studies of agile software development: A systematic review, Information and software technology 50.9 (2008): 833-859, ELSEVIER.
[10] S. Nerur, R. Mahapatra, G. Mangalaraj, Challenges of migrating to agile methodologies, Communications of the ACM (May) (2005) 72–78.
[11] Mahizharuvi and Alagarsamy, A Security Approach in System Development Life Cycle, Int. J. Comp. Tech., Vol. 2 (2), 253-257, 2011.
[12] Devanbu, Premkumar T., and Stuart Stubblebine. "Software engineering for security: a roadmap", Proceedings of the Conference on the Future of Software Engineering, ACM, 2000.
[13] Dwivedi Himanshu, Mobile Application Security, Tata-McGraw Hill Education, 2010.
[14] Mouratidis, Haralambos, Paolo Giorgini, and Gordon Manson, When security meets software engineering: a case of modelling secure information systems. Information Systems 30.8 (2005): 609-629.

AUTHOR'S PROFILE

M. Bhanu Sridhar
is an Associate Professor in GVP College of Engineering for Women, Visakhapatnam, AP, India. He received his Bachelor and Master degrees in Computer Science and Engineering from Andhra University, Visakhapatnam. He completed his Doctorate from JNT University, Kakinada with specialization in Software Engineering.
His primary research interests include software reuse, testing, models and data mining..

He is a member of CSTA, IACSIT, IAENG and IRED. He is also in the review board of many journals. He published nearly 15 papers in different journals and conferences.