1. LITERATURE REVIEW

It is observed that the most of the research were focused on integral computer machines relevant software engineering concepts. They highlighted on structural part of the process methods, frame work and its significant methodology. Recently, research mentioned in future enhancement for the development of product to use the art of clean room software engineering concepts. The technique for software development and its associated technology are used for the restricted purpose and have its own limitations. Process models, frame work, project planning, metrics etc. are the parameters used to produce client satisfaction software products. This technique is just used in software development only. But it may also be useful for external study field. Referred the various paper published with the relevant subject on IEEE internet websites, Magazines, Journals, Reserve Bank of India magazine, RBI reports and journals, relevant conference proceedings, books on Software Engineering, computer Science, Information technologies and urban Co-operative bank schedules. Followings are the literature review presented here for the development of research subject.

Thomas, R (1994)

The author expressed the views on software crisis. The “software crisis” was first acknowledged in the mid 1960s. Some would claim that the software crisis is still with us. A recent issue of Scientific American ran a story on the currently infamous software disaster at the new Denver international airport. Software engineering has been heralded as the needed disciplines that can improve software development to meet the needs of modern software requirements.

Hankley, B (1995)

As per the opinion of author, software based products are an ever increasing portion of industrial output. Further suggested that, few engineers who produce software have had formal training in
the specification, design, implementation, documentation, and maintenance of large software systems. In addition, few have had training in software engineering management and development tools and processes.

**Keller, T. (1997)**

It is observed that, the Space Shuttle avionics software represents a successful integration of many of the computer industry's most advanced software engineering practices and approaches. Beginning in the late 1970s this software development and maintenance project has evolved one of the world's most mature software processes applying the principles of the highest levels of the Carnegie Mellon University Software Engineering Institute's Capability Maturity Model, Trusted Software Methodology, and ISO 9001 standards.

**McCauley, R. (1998)**

In the fall of 1994, author reorganized the content of our three-course computer science introductory sequence in order to introduce software engineering concepts early and provide a consistent software engineering focus from one course to the next. Author also established documentation and design standards that would serve as a framework for teaching the software engineering principles and techniques that we considered appropriate and essential to novice software developers.


Author observed, component-based enterprise software engineering (CBESE) is a rapidly emerging trend in the software engineering area. In the CBESE approach, software systems are no longer built from scratch. Instead, reusable software components, built by in-house developers or commercial vendors, are used as the building blocks of new component-based enterprise software systems.
Chung-Horng Lung (2000)

Software architectures are often claimed to be robust. However, there is no explicit and concrete definition of software architecture robustness. In this paper explained a definition of software architecture robustness and presents a set of architecture metrics that were applied to real-time telecommunications software for the evaluation of robustness.

Michael N. Huhns (2001)

This paper describes a new approach to the production of robust software. We first motivate the approach by explaining why the two major goals of software engineering — correct software and reusable software — are not being addressed by the current state of software practice.

C. M. Woodside (2001)

It is observed by the author, performance is determined by a system's resources and its workload. Some of the resources are software resources which are an aspect of the software architecture; some of them are even created by the software behavior. In this paper describes software resources and resource architecture, and shows how resource architecture can be determined from software architecture and behavior.

Len Tiu Wright (2002)

Author expressed his views with reference to IT industry. In the context of IT industry’s phenomenal growth and the era of New Marketing, it is noted that care is being taken by many of the leading IT Companies (including those exclusively marketing directly to their customers) to nurture their corporate brand identities.
Clarke, J (2003)

Author expressed views on metaheuristic techniques. A techniques such as genetic algorithms simulated annealing and tabu search have found wide application in most areas of engineering. These techniques have also been applied in business, financial and economic modeling. Metaheuristics have been applied to three areas of software engineering: test data generation, module clustering and cost/effort prediction, yet there remain many software engineering problems which have yet to be tackled using metaheuristics.


This paper discussed on the various parameters which affect the complexity of software system. According to the author, the complexity of software systems leads to the creation of many related diagrams, representing different viewpoints, different levels of abstraction, and different implementation alternatives. Model-driven software engineering places these models in the center of the development process. Tool support is an essential aspect of model-driven engineering.


The main focused on basic techniques of software engineering like, is software development really a form of engineering? Or is it just some kind of elaborate craftsmanship? Are we just fooling ourselves thinking that we are doing engineering? If so, it is certainly not from lack of trying hard over the last 20 years. But maybe we tackled the problem from the wrong end: we tried to impose techniques from other engineering disciplines onto software development models without understanding the real nature of software.

The main intention of the author in this paper to express views for verification and validation of design models. A unified paradigm for the verification and validation of software and systems engineering design models expressed in UML 2.0 or SysML. This paradigm relies on an established synergy between three salient approaches, which are model-checking, program analysis, and software engineering techniques.


The main intention of this paper, to express views for verification and validation of design models. A unified paradigm for the verification and validation of software and systems engineering design models expressed in UML 2.0 or SysML. This paradigm relies on an established synergy between three salient approaches, which are model-checking, program analysis, and software engineering techniques.


The main intention of the author in this paper, to express views on verification and validation of design models. A unified paradigm for the verification and validation of software and systems engineering design models expressed in UML 2.0 or SysML. This paradigm relies on an established synergy between three salient approaches, which are model-checking, program analysis, and software engineering techniques.

Jackson, M. (2007)

Developing a software-intensive system is engineering in the traditional sense: creating an artifact which transforms the physical world to meet some recognised need. The artifact is the hardware-software machine; the physical world is the system's environment; and the recognised
need is the requirement. For a successful development the entailment must hold: "machine, environment, and requirement".

**Dillon, T.S. (2008)**

This paper describes the use of ontology’s in different aspects of software engineering. Use of ontologies varies from support for software developers at multiple sites to the use of ontology to provide semantics in different categories of software, particularly on the Web. The world's first and only software engineering ontology and project management ontology in conjunction with domain ontology are used to provide support for software development that is taking place at multiple sites.

**Dongsun Kim (2008)**

Author proposed the educational toolkit focusing on architecture design methodology for embedded software and reports experience with teaching software engineering by using the toolkit. The toolkit consists of three perspectives: people, process, and technology. Each perspective represents a set of tools which can support educational activities.

**Ellis, H (2008)**

Software engineering has emerged as a discipline, but it is still new and relatively small. It is important to consider how this process has progressed, and to ensure that the continued evolution of the discipline has the support and attention that will ensure success.

**Mannaert, H. (2008)**

Unlike other engineering areas, software engineering has yet to produce designs that enable automated production. This seems related to the fact that the fundamental laws for software engineering have yet to be found, leading to the belief that researchers should strive for a unified theory of software engineering.
Lago, P. (2009)

Software engineering courses often include some form of project, aimed at bridging the theory-practice gap. These projects tend to emphasize technical topics. Because software engineering has an important organizational and social dimension as well, and because software engineering courses may be taken by non-CS majors too, there is every reason to also include these non-technical issues in the software project.


According to author, as businesses rely on software solutions to preserve their position in a highly competitive market, the need for reliable and robust software systems is vital. Lately, there has been a significant interest in building software using models as their main artifacts. Unlike traditional development techniques which tend to be code-centric, model driven approaches, such as the Model Driven Architecture (MDA) standard, stress the usage of models at all levels of the software development life-cycle.

Iqbal, M (2009)

As per the author(s) view, the results (efficient and improved software process models) will be implemented in the software industry, by software project managers, by focusing on the list of activities giving 80 percent of the overall productivity of the software process. The results will be the list of activities having the capability to give 20 percent of the overall productivity of the software Waterfall process, to be ignored, eliminated or designated to reduce the effort.

Dakhli, S. (2009)

Many papers proposed in the software engineering and information systems literature are dedicated to analysis of software projects missing their schedules, exceeding their budgets, delivering software products with poor quality and in some cases even wrong functionality.
Various solutions have been proposed by academics and practitioners in order to deal with the software crisis, counter these trends and improve productivity and software quality.

**A.M. Nageswara (2009)**

In view of software development, software effort estimation plays an important role in determining the success or failure of a software project. The feasibility study for taking up a software project must include effort required to design and develop the project and schedule. These parameters will give an idea of the cost involved in taking up such projects. Therefore, it is crucial that effort estimates are reliable. Unrealistic estimates will lead to disastrous consequences for both the developer and user.

**Mirian-Hosseinabadi, S.H (2010)**

As per the author(s) opinion, using software products as engineering artifacts is becoming more widespread in industry. It leads to increasing the need for training students and experts in the academic centers with software developing skills in accordance with the engineering principles and disciplines. However, one of the main challenges in the software engineering education is how to teach engineering disciplines to the students so that we can inspire the engineering concept to them.

**Fendler, J. (2010)**

The discipline of Software Engineering is continuously adapting to new challenges while gaining more and more insights. The age of globalisation has brought about a new movement of internationalization and localisation. While practitioners fully embrace the efforts, educators only marginally consider the implications for the teaching and learning of Software Engineering.

Literature survey is covering the theory of Information technologies, co-operative financial sectors, and similar type of projects, study reports. Visited subject related internet sites and referred the various books for preparation of basic model for the study work. Conceptual
literature consisting theories and various concepts. Personal discussions have been done with the bank management, officers, employees, customers, vendors, Information Technologies experts and related people etc.

It has found that in most of the significant research papers, software engineering methodology are going to use in latest applications though it has some challenges and it requires more research in that.

Josh Dehlinger [27] suggested adoption of advanced application with respect to software engineering methodology. The rapid proliferation and ubiquity of mobile, smart devices in the consumer market has forced the software engineering community to quickly adapt development approaches conscious of the novel capabilities of mobile applications. The combination of computing power, access to novel onboard sensors and ease of application transfer to market has made mobile devices the new computing platform for businesses and independent developers. However, the growth of this new computing platform has outpaced the software engineering work tailored to mobile application development.

Barbara G. Ryder [7] studied the programming concepts for software development. The production of software requires software engineering techniques, such as specification, design, implementation, testing, and maintenance. Essential to performing the last three phases of software development is the selection of a programming language(s) as an implementation vehicle. The programming language(s) chosen needs to offer the application programmer the power to naturally express the task at hand in a disciplined manner.

Thomas Østerlie [49] suggested software engineering (SE) publications have been a major channel for OSSD (open source software development) research. In the SE research literature on
OSSD, it has found that the more research requires for achieving clear views and further improvement in OSSD.

Most of the literature containing the following type of theory which will be useful to the proposed study work –

**N. Ramu (2009)**

The author explained the status of first UCB in Tamil Nadu. The Urban Cooperative Banking (UCB) system has come a long way since 1904 when the first UCB was started at Kancheepuram in Tamil Nadu. UCBs remain not-for-profit, owned and controlled by the members who use their services. They are unit banks of the American model rather than branch banks of the British model.

**N Ganesan (2009)**

Author examined the efficiency of 30 State Cooperative Banks (SCBs) and 20 District Central Cooperative Banks (DCCBs) in India during the period 2002-06. Author emphasizes on the performance of SCBs and DCCBs in terms of its technical efficient score. In recent years, financial institutions, especially banks, have had to operate in an increasingly competitive environment under financial sector reforms, due to open and global economy.

**Amit Basak (2009)**

The author mainly focused on efficiency in UCBs. He explained his views with respect to Performance Appraisal of Urban Cooperative Banks. Urban Cooperative Banks (UCBs) figure among the vital segments of the banking industry of the country. They essentially cater to the credit needs of persons of small means. Though some UCBs have performed creditably in the recent years, a large number of them have shown discernible signs of weakness.