Abstract

The objective of this research was to study, categorize and examine various methods for describing or modeling, as how software systems are developed and to provide solutions in the form of a generalized model which help in producing a correct and high quality software for the companies.

A study and analysis of traditional software life cycle models that dominate the field and current software development practices is conducted. It is followed by a more comprehensive review of the traditional models of software evolution that are of significant use and are considered as the basis for organizing software engineering projects and technologies. The idea of a software process model that fits every project seems unlikely because every project has so many aspects that it is difficult to capture every potential aspect in a single perspective, the study helped to provide solutions in the form of a generalized model which may help in producing a correct and high quality software for the companies.

Firstly the concept of software engineering is explained, it’s primary role as a part of software development process and it’s purpose in the field of software development is discussed. It is followed by the second objective to explain the software process as a systematic and disciplined approach to the development and maintenance of the software and the methodology it uses to provide systematic, cost effective and reliable way to develop software. Next, the third objective is explained and describes the basic activities for the development of software, which are used in all the traditional software process models, which serve as the basis of any new process model developed. Lastly the fourth objective to explain software process modeling and its aim along with the advantages of implementing software engineering process models is given.
A general evaluation objective, which means to appraise or assess the significance of evaluation, is presented. It includes the general evaluation objectives, which take into account a general understanding of the current practices, confirming the theories or conventional knowledge, exploring when a domain is not well understood and describing the current state of things. Evaluation helps in predicting the future and explaining why the things or sequences are taking place.

In software engineering, for evaluation measures we need to understand the software process and the product. Various approaches to perform the software evaluation are discussed in this chapter. Methods for model definition are also discussed. It explains various types of customers for whom the software is developed followed by the methods for data capture. The existing methods for analysis are also described. Further, it discusses the verification and validation techniques for managing the validity treats. The research methods designed for objective evaluation are stated considering that evaluation can be conducted using construct validity, internal validity, external validity and reliability.

In the next section the evaluation process is covered which consists of three stages namely, the quality refinement definition, evaluation preparation and evaluation procedure. Next, methods for solving a model are described which includes how to solve a model analytically, where a complete mathematical description of the model is required. If an analytical solution of the model is not feasible due to the limitations of the modeling formalism or model complexity, or not desirable, another way, by means of using computer based simulation is described in which one may approximate the behavior of the real system by executing the model over time, and subsequently or interactively draw conclusions about reality from the observed dynamic behavior of the model.

Then the software process is revisited from the point of view that there exists a need for a generalized model after studying the basis of process model development. The
factors that influence the process models and enhance the capabilities of the processes are described. The fundamental design concepts, which have evolved, are discussed here.

Structured planning and execution of the activities is required for model development and simulation. It is called the Model Development and Execution Process, which is described in the subsequent section. Modeling activities and intermediate products describe the activities carried out during model development, which can be organized in phases, according to their contents and chronological occurrence. Numerous model development processes exist, which can be distinguished by the selection of phases and the way in which the feedback between the phases is organized. It is also discussed that not only the iteration of development phases, but also the resolution or level of detail of the activity descriptions making up each phase can vary and it mainly depends on the architect of the model development process and the aspects of model development that seem to be most important to the particular group. Several classes of process models for the development of models, software products, or simulation results are studied, namely, Waterfall processes, V-Type processes, Lifecycle processes and Spiral processes. It is followed by design considerations. There are many aspects to consider in the design of software. The model design should be strong enough to contain the basic considerations while developing a system. The importance of each aspect should be taken care of which would reflect the goals the software that are tried to achieve.

In subsequent section Software Design Description are described and also the IEEE 1016-1998, which is also known as the Recommended Practice for Software Design Descriptions. This IEEE standard specifies an organizational structure for a Software Design Description (SDD).

We reviewed the literature and presented a brief summary of the work relating to the general process model. Here, we describe why we need the GSPM- Generalized
Software Process Model. It affirms that the need to obtain a general model arises as the language independent process can be established and the techniques can then be designed that guarantee the conservation of model correctness during configuration, thus liberating analysts from using manual methods to check the model correctness. Model should be able to eliminate redundancies in the process and facilitate standardization and reuse. Thus it is tried to develop an integrated framework or system to manage variability in process aware information system.

Study and comparison of the process models are performed to ensure the need for obtaining a generalized model. The traditional software lifecycle models are described and also as to where such models are needed, that is, in which situation such models are beneficial. A comparison table is also drawn for the traditional models.

Next, the shortcomings of various models are discussed. This is followed by an efficient paradigm for implementation of process models in projects. Initially, a model has been designed and proposed for the projects with flexible requirements. This is tested by showing the implementation in of the proposed model in the courseware development activity / project. Followed by is a new proposal that has been developed for the projects with dynamic and collaborative requirements. This model is further evaluated for its functioning.

The proposed work with the new concepts is presented, so as to eliminate the shortcomings and introduce new method for taking decisions. The Generalized Model design is discussed and presented which is modeled for the optimization of the process, which will help the software development team to work with larger efficiency and effectiveness. It is followed by the basic structural model designed for it. Next is the data flow diagram of the GSPM followed by the internal structure of GSPM. For a process model to be successful it is essential that confirmation of the activities is performed along with the participation of the stakeholders. Once the confirmation is sought the activity of code design and development should take place.
The re-use of the components acts as the basis of reducing time factor in the development phase. The organization of the module should be such that it should be easy to modify. Execution of the process should be easy to measure and it should be possible to refine that execution. Model should be flexible so that it can provide guidelines on the execution and tracking of a project. Thus emphasis has been on productivity improvement and reuse building. Effective testing and continuous validation from customer side will enhance confidence in services being provided making the system more stable.

The case study performed on the e governance is described and the benefits are listed. Another case study is performed on Student Information System, a project carried out at Birla Institute of Technology, Mesra, Ranchi, Extension Centre, Jaipur. The comparison is made between the project using the mostly used Spiral methodology and the proposed methodology, GSPM. Thereby depicting the effectiveness and usefulness of the proposed model.

Finally, based on the case study, an analysis is drawn using the mathematical modeling language, Petri net, which is a directed bipartite graph offering graphical notation for stepwise processes. Bar charts indicating the number of inputs, number of function points and number of iterations are drawn for easy comparison of the existing and the proposed Generalized Software Process Model [GSPM]. This states the advantages of using the proposed model. We demonstrate the applicability of the proposed approach by using Petri nets for all the traditional models and the proposed model. It is followed by the graphical analysis based on the number of input points, number of function points and number of iterations, which are the basis of model execution and performance checking. Time, effort and team size are also considered to prove the model’s importance.