The Pros and Cons of the Most Utilized Software Development Models

Mohamed S. M. Hasni
Kastamonu University, Institute of Science
Dept. of Mathematics, Turkey
Email: mhasni@ogr.kastamonu.edu.tr

Turhan Köprübaşi
Kastamonu University, Institute of Science
Dept. of Material Science and Engineering, Turkey
Email: tkoprubasi@kastamonu.edu.tr

Abstract—
This paper explores different models of software development based on their definition and process. Moreover, the models are connected to the lifecycle of the software project. The different software development models differ based on their approach to the requirements and the end product. The risk associated with each model is one of the distinctive features of each one of them. The study provides a comparison based on the advantages and disadvantages of each development model, which allows researchers and practitioners to make a choice in their research or project.

Keywords—Software Development Models, comparative study, software lifecycle

I. INTRODUCTION
In software development there are different types of strategies that are followed to achieve the goals of the projects. The main goals which always concern software project managers are cost, time, quality and intellectual property protection [1]. Therefore, this paper focuses on the main development strategies that are followed by software developers, which aims to understand the scope, management tools, advantages and challenges that are involved. Furthermore, the paper provides a comparison between different software development models to form an understanding of their similarities and differences.

II. SOFTWARE DEVELOPMENT MODELS
A software is defined as a command or set of commands that are able to carry out specific functions in order to work in the purpose it is designed for, where the data structure enables the software to operate them, and document and evaluate information according to their usability and importance. Other definitions describe the software as a set of designed instructions that operate on command in order to produce a desired result. The software shall be maintainable, dependable, efficient and usable in order to be considered decent [2]. There are five main phases that describe the lifecycle of software development, as shown in Figure 1:

- Software planning: where the team intended to build the software is structured, the scope and purpose of the software is determined, issues with current solutions and potential future problems are identified, and technology priority and market are determined.
- Software analysis: the current technology and literature are surveyed in order to understand the full scope of the existing technology and the potential problem that the intended software is aiming to solve, modelling the new software based on brainstorming sessions among the developer’s team, categorizing the issues, solutions and opportunities which the new software is creating, and analyzing for the requirements of the software development.
- Software design: constructing the functions that form the software along with the interaction results, compiling the database schema, and building a platform for user interface.
- Software implementation: based on the design scheme, the database is created. Subsequently, software applications are structured based on the software design, as well as testing the software functions and fixing any possible bugs.
Software maintenance: through a continuous research, testing and customer feedback, the developer shall be able to fix any bugs and enhance the software to meet the objectives [2].

In the software development process, there are several factors that affect the choice of the process and strategy used, which include project type and size, development duration, software and development complexity, risks that are associated with the development process in terms of type and level, user requirements, application areas, level of customer involvement, developer’s level of experience, available team size and skill pool, the level of interaction between the software, developer and user, and the availability of the technology to develop the desired software. Based on their factors, the software developer is able to choose a development model that suits them; however, the most common used software development models are as the following [3,4]:

- Waterfall model: the simplest development model, where the activities in software development are subsequent to each other in a linear form. The majority of the software projects that use this model are not concerned with changing requirement during the development.
- V-Shaped Model: the software development phases are constructed in a V-shaped model instead of linear, where planning and maintenance are worked side by side with the maintenance phase, specifications are coupled with testing, and design is coupled with unit and integration testing, as shown in Figure 2.
- Prototype model: it is a development strategy, where the developer builds an incomplete version of the software in order to create and initial final vision of the required software. This model is used to bridge any gaps between the expectation of the customer and the vision and understanding of the developer. Moreover, the main aim of this model is to reduce iterations and adopt a more flexible development process.
- Spiral model: a model used for complex and expensive software in order to combine the waterfall and prototype models by investing in their advantages. The phases of the spiral model are similar to the waterfall model; however, the sub-phases of planning, risk evaluation and prototyping are added in between in order to achieve the model aims.
- Incremental model: this method may involve the same phases used in the waterfall and prototype models; however, the software is developed in portions rather than a complete entity.
- Extreme (Agile) model: this method involves several teams, in addition to the customer, in a more complex interaction between the different team parts, where time is limited by the customer and the requirements are ambiguous [5,6].

![Fig. 1: Software development lifecycle [2]](image)
Based on studying the a few software development models, a comparison between four models; waterfall, V-Shaped, Spiral and Extreme is performed, as shown in Table I. The comparison shows that the differences between the models are found in different aspects. Models differ based on the complexity of the project and the easiness to understand its requirements. Moreover, the project’s size plays a major role in choosing the most suitable model. Some of the models are rigid and do not allow changes during the development process, which causes issues to be discovered at the end of the project, where changes would eventually cause delays and increase in project’s costs.

Project management is affected by the chosen software development models, where a model like the extreme model depends mainly on documenting the workflows and processes during the development, which makes its implementation difficult in larger projects. Moreover, as some models do not allow iterations or testing the software during the development stages, it becomes difficult to implement these models as the complexity of the project requirements increase.

### Table I Comparison of Pros and Cons of Software Development Models

<table>
<thead>
<tr>
<th>Software Development Model</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
</table>
| Waterfall                   | - Easy implementation and understanding.  
- Used in a lot of projects.  
- Insistence on definition before tasks.  
- Deliverables and mile stones are identified.  
- Used in the case of mature products and weak teams.  
| Mismatched with reality situations.  
- No iterations.  
- Not for accurate requirements.  
- Serious issues are discovered at the end.  
- Risk management cannot be performed.  
- Changes are difficult and expensive.  
- Costly overhead.  |
| V-Shaped                    | - Simplicity and ease of use.  
- Deliverables are based on each phase.  
- Success rate is higher than waterfall model due to presence of test plans.  
- Suitable for small projects with simple requirements.  
| No flexibility, similar to the waterfall model.  
- Scope adjustments are difficult and costly.  
- The software is development through implementation.  
- No solutions for arising issues in the testing phase.  |
| Spiral                      | - Risk is well analysed.  
- Suitable for large projects with critical  | Likely to increase costs.  
- High expertise for risk analysis.  |
IV. CONCLUSIONS

This study provides a review of several software development models and the development processes they are designed with. Therefore, it shows from the comparison section of this paper that the choice of a software development model depends on few factors, including project size and complexity. It is important to understand the success of each model and its association with the characteristics of the project. The study provides a comparison based on advantages and disadvantages, which allows researchers and practitioners to form an understanding of the direction they choose for their projects.

REFERENCES


