Addis Ababa Science and Technology University

College of Engineering

Department of Software Engineering

Software Component Design(SWEG5107)

Group Members	ID
1. Mahlet Hailu	ETS0794/13
2. Naod Ararsa	ETS0956/13
3. Naod Zekaria	ETS0961/13
4. Naol Yadete	ETS0968/13
5. Natnael Tafesse	ETS1026/13

Introduction

Software Development Life Cycle (SDLC Models) are the basic building blocks upon which any software product is developed. These models define the overall structure of any software product.

The objective of SDLC Models is to generate high-quality software products keeping the requirement, cost, time, and expectation in mind.

There are several SDLC Models available, each one having its specialties and abilities. These models are used according to the needs and requirements of software products. Some of the models are only used for specific types of software products and some might be used for any type of software product.

The reason behind using the models for the development rather than developing the whole software product at one go is that each and every phase in developing the software is clearly designed and mentioned in these SDLC Models. This helps in better judgment, cost-effectiveness, and successful development of software products.

Build & Fix Model

The **Build and Fix Model** is a traditional software development methodology that focuses on creating a working version of the software as quickly as possible and then repeatedly modifying it until it meets the requirements or is deemed satisfactory. It is a simple and unstructured approach and is often considered the earliest method of software development.
The build and fix model is an approach and model used for building a software product, consisting of only two phases.

Phases of the Build and Fix Model

1. Build Phase

In this phase, the developer starts by writing the initial version of the software with minimal planning or requirements analysis. The focus is on quickly creating a working product that provides basic functionality. This emphasis on rapid development often results in a product that lacks comprehensive design and documentation.

Key Characteristics:

- Minimal or no documentation.
- Rapid prototyping with little attention to design principles.
- Immediate implementation of user requirements or developer ideas.

2. Fix Phase

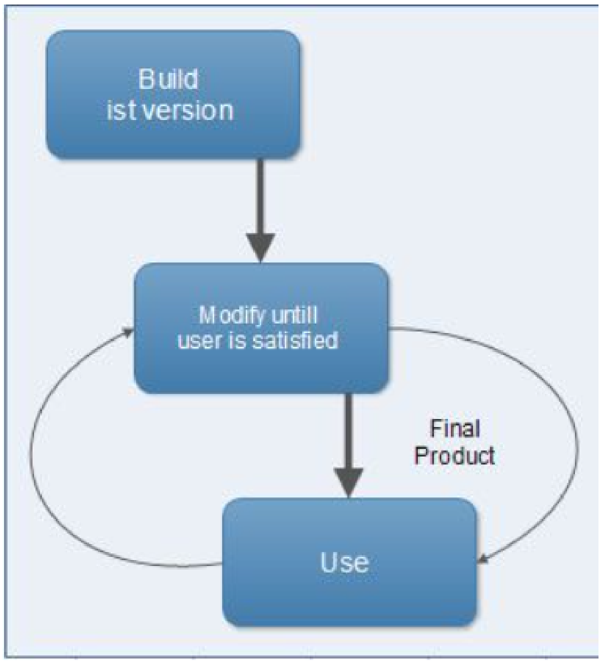
After the initial build, the product undergoes repeated modifications based on feedback, errors, or newly discovered requirements. Developers fix bugs, add features, and refine the software iteratively.

Key Characteristics:

- Ad-hoc adjustments and fixes.
- Iterative process without a predefined number of cycles.

- Constant response to emerging issues and user feedback.

Once developed, it is handed over to the client without any testing possibilities. The client checks the software functions, features and behavior. Upon which if any feature or function does not performs properly, the developer needs to make the changes again.
This process goes on and on until the client is completely satisfied with the developed software product. But with repeated changes the software, it might become impossible to make changes in the code causing more problems.



Hence, it is not advisable to use Build & Fix model for projects of larger sizes as maintenance is not possible if this model is used.

Advantages of the Build and Fix Model

1. **Simplicity**: Easy to understand and implement, requiring no formal training or processes.
2. **Quick Start**: Development can begin almost immediately without extensive planning.
3. **Adaptability**: The iterative nature allows for quick adjustments based on user feedback or requirements.

4. **Low Initial Investment**: Minimal planning reduces initial time and cost.

Disadvantages of the Build and Fix Model

1. **Lack of Structure**: Absence of a defined process can lead to chaotic development.
2. **Cost Overruns**: Repeated fixes and changes may significantly increase costs over time.
3. **Poor Scalability**: Difficult to maintain or scale due to lack of documentation and design considerations.
4. **Quality Issues**: Limited focus on design, testing, and quality assurance can result in a subpar product.
5. **Unpredictable Timeline**: The iterative process can extend indefinitely, making it hard to estimate project duration.

Limitations of the Build and Fix Model

- Not suitable for complex or large-scale projects due to its lack of planning and structure.
- High risk of technical debt as the product evolves without a robust design.
- Difficulties in maintaining and upgrading the software due to poor documentation.
- Limited applicability in projects requiring compliance with formal standards or regulations.

When to Use and When Not to Use the Build and Fix Model

When to Use:

- For small, simple projects with minimal requirements.
- When the timeline is extremely short, and delivering a functional product quickly is the primary goal.
- In scenarios where formal processes are not feasible or cost-effective.

When Not to Use:

- For large, complex, or mission-critical systems where scalability, reliability, and maintainability are essential.
- When the project requires thorough documentation, rigorous testing, or compliance with industry standards.
- In teams or organizations that prioritize structured methodologies and predictable timelines.

Conclusion

In summary, the Build and Fix Model is a basic approach to software development that works best for small or experimental projects. While it's easy to understand and get started with, it lacks structure and can lead to problems with scaling, maintenance, and quality. This makes it unsuitable for bigger or more critical projects. Developers should think carefully about what they need, the risks involved, and the project's limits before using this model to make sure it fits their goals and resources.