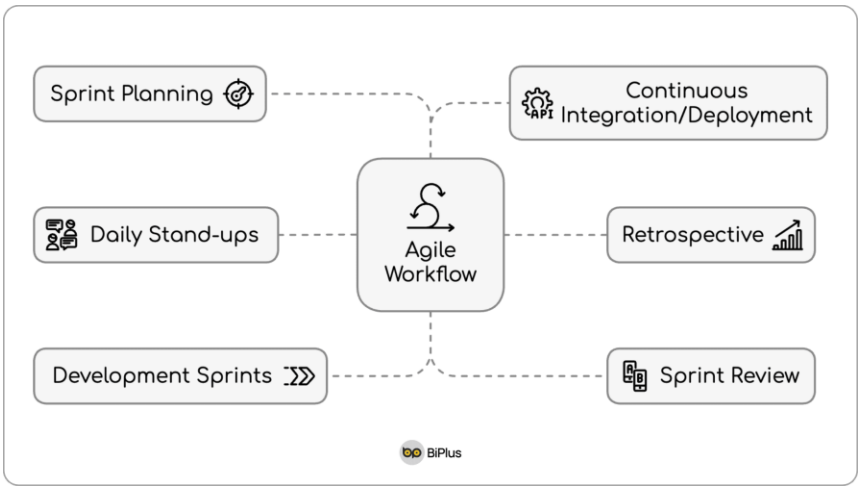
1.7.5 Agile Model

1.7.5.1 Definition

The Agile model emphasizes iterative development, where requirements and solutions evolve through collaboration between cross-functional teams. It focuses on flexibility, customer feedback, and delivering small increments of the product.



1.7.5.2 Usage Situations

- Dynamic and rapidly changing requirements.
- Projects requiring frequent updates and user involvement.
- Software needing early market entry with incremental updates.

1.7.5.3 Advantages

- High customer satisfaction through regular feedback.
- Early delivery of functional modules.
- Flexibility to accommodate changes.

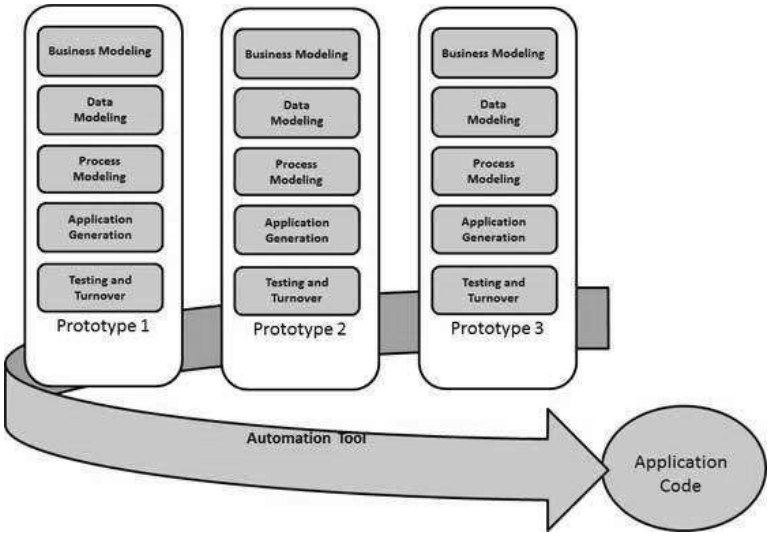
1.7.5.4 Disadvantages

- Requires active user involvement, which may be resource-intensive.
- Not ideal for projects with fixed scope and budget.
- Difficult to predict costs and timelines.

1.7.6 RAD (Rapid Application Development) Model

1.7.6.1 Definition

The RAD model is a type of incremental model that focuses on rapid prototyping and quick delivery of the software product with user feedback at each stage.



1.7.6.2 Usage Situations

- When time is a critical factor (short deadlines).
- Projects with clearly defined requirements.
- Applications requiring frequent interface updates.

1.7.6.3 Advantages

- Rapid development with reduced development time.
- High customer satisfaction through active user feedback.
- Flexible and easy to accommodate changes.

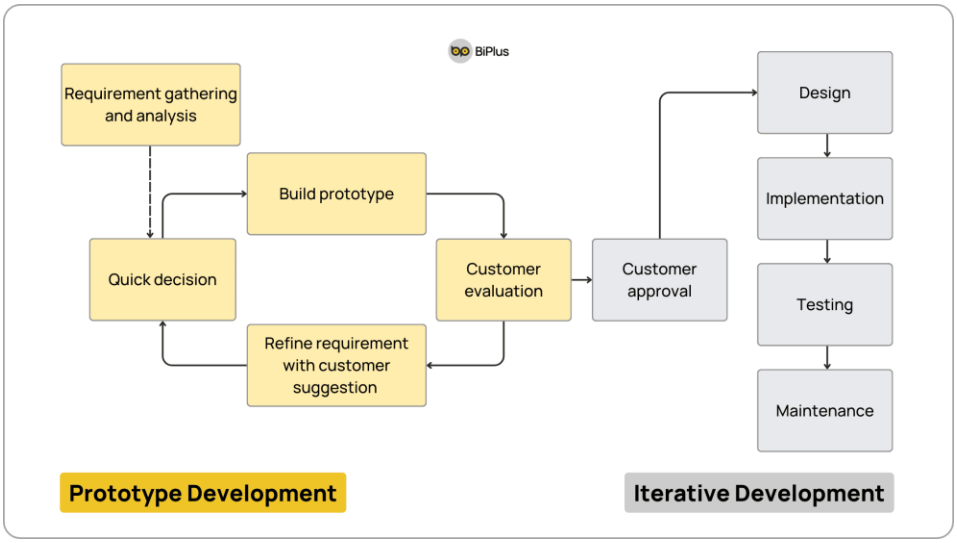
1.7.6.4 Disadvantages

- Requires highly skilled developers.
- Not suitable for large, complex systems.
- May result in compromised quality due to tight schedules.

1.7 7 Prototype Model

1.7.7.1 Definition

The Prototype model involves creating a working model of the software before the actual development begins. It allows users to visualize and refine requirements early.



1.7.7.2 Usage Situations

- Projects where requirements are unclear or not well-defined.
- When user interaction and interface design are critical.
- Applications needing high customer involvement.

1.7.7.3 Advantages

- Clarifies requirements early in the process.
- Reduces misunderstandings between users and developers.
- Identifies risks and design flaws early.

1.7.7.4 Disadvantages

- Prototyping can be time-consuming.
- May lead to scope creep if users keep changing requirements.
- Not suitable for large or highly complex systems.