The Evolution of Software Development: Methodologies and Best Practices

Yury Akuzum*

Department of Computer Science, Drexel University, Philadelphia, USA

Corresponding author: Yury Akuzum, Department of Computer Science, Drexel University, Philadelphia, USA, Email: yuryakuzum67@hotmail.com

Introduction

Software development is a dynamic and ever-evolving field that has transformed the way we interact with technology. Over the years, software development methodologies and best practices have evolved to enhance efficiency, collaboration, and quality in the development process. This research article explores the evolution of software development, discusses current methodologies used in the industry, and highlights best practices for successful software development. The Waterfall model, developed in the 1970s, follows a sequential development process. Each phase, such as requirements gathering, design, implementation, testing, and maintenance, is completed before moving on to the next phase. This approach emphasizes extensive documentation and a structured development lifecycle. The Agile Manifesto, introduced in 2001, emphasized iterative and incremental development. Agile methodologies focus on adaptability, collaboration, and delivering customer value. The Scrum framework, with its time-boxed iterations (sprints) and cross-functional teams, became popular for managing complex software projects. Kanban and Lean Development emphasize continuous improvement and waste reduction. Agile methodologies, such as Scrum, prioritize customer collaboration, flexibility, and rapid feedback. Scrum teams work in short iterations, delivering working software at the end of each sprint. Daily stand-up meetings, sprint planning, and retrospectives ensure efficient communication and continuous improvement. Agile methodologies promote adaptive planning and respond to changing customer needs.

DevOps

DevOps is a collaborative approach that integrates software development and operations. It focuses on automation, communication, and collaboration between development and operations teams. DevOps emphasizes continuous integration and Continuous Deployment (CI/CD), enabling rapid and frequent software releases. Infrastructure as Code (IaC) and configuration management tools facilitate automated infrastructure provisioning and deployment. Adopting Agile principles, such as delivering working software frequently, embracing change, and promoting collaboration, enhances project success. Teams should prioritize customer satisfaction, welcome change, and promote open communication throughout the development process. Implementing continuous integration practices ensures that code changes are integrated regularly, enabling early detection of integration issues. Automated testing, including unit testing, integration testing, and acceptance testing, provides fast and reliable feedback on the software's quality. Utilizing version control systems, such as Git, enables teams to manage code changes efficiently and collaborate effectively. Project management tools, issue tracking systems, and communication platforms facilitate seamless collaboration and project tracking. Software development methodologies and best practices have evolved significantly to accommodate changing market demands and emerging technologies. The shift from the Waterfall model to agile methodologies, such as Scrum and Kanban, has revolutionized the development process, emphasizing collaboration, adaptability, and customer value. The adoption of DevOps practices further enhances efficiency and quality by integrating development and operations.

Business process and data modelling

A very efficient method for presenting information to system developers and users is a graphic representation of the current state of the information. The organizations that carry out the functions of the business process being modeled are depicted in a business model. A foundation is built to visualize, define, comprehend, and validate the nature of a process by depicting activities and information flows. An information model gives the subtleties of data to be put away and is of essential use when the end result is the age of program code for an application or the readiness of a practical determination to help a PC programming settle on or-purchase choice. An illustration of how data models and business processes interact can be found in the figure on the right. In most cases, a business analysis, or interview, is used to create a model. A facilitator conducts the interview by asking a series of inquiries aimed at obtaining necessary process-related information. To emphasize that the participants are the ones who provide the information, the interviewer is referred to as a facilitator. The facilitator should be familiar with the process of interest, but this is less important than having a structured method for asking the process expert questions. The methodology is important because, once the information from all of the interviewers has been gathered, it needs to fit together. Typically, a team of facilitators collects the

information from across the facility. The models are created with the goal of defining either the current state of the process, which is referred to as the "as-is" snapshot model, or a collection of concepts regarding what the process ought to include, which is referred to as the "what-can-be" model. The creation of process and data models can be used to determine whether re-engineering is required as a corrective measure or if the existing processes and information systems are sound and only require minor modifications or enhancements. The production of plans of action is in excess of a method for review or computerize your data cycle. The way your business or organization runs its operations can be fundamentally changed by using analysis.