

SOFTWARE DEVELOPMENT METHODOLOGIES AND THEIR ROLE IN THE SDLC

Vinnitsia national technical university

Анотація

Дана стаття демонструє основні методології розробки програмного забезпечення та їх ролі в життєвому циклі проекту.

Ключові слова: програмне забезпечення, методології, життєвий цикл проекту, проєкт, Agile, V-model, Waterfall.

Abstract

This article demonstrates the main software development methodologies and their roles in the project life cycle.

Keywords: software, methodology, project lifecycle, project, Agile, V-model, Waterfall.

In the world of IT, new projects are launched every day, new ideas are conceived and brought to life. In order for all development processes to proceed as quickly and efficiently as possible, various development methodologies are used, depending on the objectives. The development, design, and planning of high-quality software is also governed by a separate process called SDLC or Software Development Lifecycle.

SDLC is divided into six main stages: planning, analysis, design, development, pilot operation, and system support [1]. Each of these stages is important during software development, because skipping one of the previous stages will cause problems in the following ones.

In the first phase, the planning stage, it is precisely decided what exactly will be done, what problems will be solved, and what needs will be met by the product. It is at this stage that the development methodology is chosen.

The first and oldest methodology is Waterfall. It is considered traditional and allows the process to be presented as a linear flow with a given sequence so that users understand that the next stage comes after the previous one has been completed. According to this methodology, developers cannot return to the previous stage, and each subsequent stage begins after the previous one is completed. This model is suitable for simple and stable projects where software requirements are clearly defined and do not change throughout the development process. For effective and coordinated work on a project using this model, it is necessary to clearly plan the schedule of work and resources.

The next methodology is Agile. It is a flexible development methodology that involves constant testing and adaptation, teamwork, self-organization, and accountability. It allows you to change the work plan or certain parts of the project during development. Developers work in so-called sprints – periods of time ranging from one week to one month. Agile has two frameworks: Scrum and Kanban. They differ in that Scrum divides the work into short periods of time, 1-2 weeks each, while Kanban has less strict restrictions, without rigid sprints. Generally speaking, Scrum is suitable for managing a single large-scale project lasting three months or more, while Kanban is suitable for small projects that do not require much time for planning, or for large, long-term projects where precise specifications are not important and tasks are formulated during the development process.

The last development model mentioned here is the V-model. The structure of this model resembles the letter V, hence its name. It is an improved version of the classic waterfall model. At each stage, the current process is monitored to ensure that it is possible to move on to the next level. Another feature of this model is that testing begins at the very beginning, i.e., at the requirements writing stage. Each subsequent stage has its own level of test coverage, and a separate test plan is developed for each level of testing. That is why, from the testers' point of view, this model is better than others, because at each stage, testing strategies for the next stage are developed, which is very important for high-quality and effective testing.

After the planning phase, when there is already a clear plan and the development methodology has been selected, they move on to the second phase – the analysis phase. At this stage, work with documents takes place. It is necessary to understand and document user requirements, determine their expectations and how to

meet them. After that, as a result of a clear understanding of the requirements, an SRS document is created. This document must be fully understood by the development team and reviewed by the customer for future use.

The next phase is design or engineering. It begins when there is a clear understanding of user requirements and the team knows exactly what needs to be implemented. This phase determines how the system will look and function, what its elements, components, architecture, etc. will be. At the end, an extended design is made, taking into account all technical and functional requirements.

The fourth phase is development. Once the requirements and specifications are fully understood, the implementation of all software components begins. This phase also includes implementation and deployment, which means that at this stage, the system will be completely ready for installation at the customer's site and for launch into operational mode.

Pilot operation and integration is the fifth phase of the project life cycle. Here, various components and subsystems are combined into a single integrated system. Various input data and analysis of output, behaviour, and functioning are fed into the system. The first User Acceptance Testing takes place, for which a copy close to the product is created, and the customer, together with interested parties, perform testing. If they are satisfied, they give their consent to the release of the product.

The final phase in SDLC is system support. This includes periodic technical support for the system to ensure that it does not become obsolete, replacement of old equipment, performance evaluation, and improvement of certain components to ensure that the system complies with standards and the latest technologies.

Compliance with all phases of the SDLC is a key parameter in software development, since the absence of one of them will make the development process less clear and coordinated, and a correctly chosen methodology is an important tool that ensures that the system will be developed on time, within budget, and will meet user needs.

REFERENCES

1. 6 main stages of SDLC. Available at: <https://qagroup.com.ua/publications/6-osnovnykh-etapiv-sdlc/>.
2. SDLC – Software Development Lifecycle. Available at: <https://www.it-notes.wiki/other/software-development-lifecycle/>.
3. 12 best software development methodologies with advantages and disadvantages. Available at: <https://www.smart-it.com/uk/2021/08/12-best-software-development-methodologies-with-pros-and-cons/>.
4. Flexible software development methodology – Agile. Available at: <https://training.qatestlab.com/blog/technical-articles/flexible-software-development-methodology-agile/>.
5. Key software development methodologies: teamwork from within. Available at: <https://wezom.com.ua/ua/blog/metodologija-razrabotki-programmnogo-obespechenija>.

Busygina Veronica – a student of Faculty of Intelligent Information Technologies and Automation, 1 PKT-24b group, Vinnytsia National Technical University, Vinnytsia. E-mail: veronikabysygina@gmail.com.

Scientific supervisor Kriutchenko Olena Oleksandrivna – a teacher of English, a Department of the Foreign Languages, Vinnytsia National Technical University. E-mail: olena.kriut@gmail.com.

Бусигіна Вероніка Павлівна – студентка групи 1 ПКТ-24б, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця. E-mail: veronikabysygina@gmail.com.

Науковий керівник: Крютченко Олена Олександрівна, викладач англійської мови, кафедра іноземних мов, Вінницький національний технічний університет. E-mail: olena.kriut@gmail.com.