PRINCIPLES FOR DESIGNING PEDAGOGICAL SOFTWARE TOOLS

Shamsutdinov Fazliddin Akhmadovich
Navoi State University
Fazliddinshamsutdinov07@gmail.com

Abstract. *This article examines the stages of designing and developing pedagogical software systems intended to support students' intellectual development and enhance their creative thinking within digital learning environments.*

Keywords: *Pedagogical software, information technologies, design principles, pedagogical software products, didactic tools.*

In the context of the rapidly evolving modern information technologies, it becomes essential to identify the fundamental principles that define the study of disciplines related to the design of pedagogical software tools. This includes specifying the requirements for such tools, adopting innovative approaches to applying educational methods, and clarifying the principles relied upon to enhance teaching effectiveness, while refining them where necessary.

Pedagogical software tools are didactic resources designed to partially or fully automate the educational process using information technologies. Moreover, modern information technologies are employed as instructional tools to enhance the effectiveness of teaching. The design of pedagogical software tools encompasses a range of components, including software products aimed at achieving specific didactic objectives within a discipline (a suite of project-based programs), technical and methodological support, and supplementary or auxiliary tools.

The creation of pedagogical software tools is carried out in several stages. The first stage involves pedagogical design, during which educational objectives are identified, and the content and structure of instruction are analyzed based on didactic potential. The second stage entails methodological design, in which scientific and theoretical knowledge is transformed into instructional materials, and curricula are developed. Principles for developing learning materials that align with the objectives, methods, and tasks of education are established, while areas for the use of electronic educational resources and distance learning are determined.

In this regard, within the scope of the study, a set of principles guiding the design of pedagogical software tools was developed:

1.Modularity Principle. A pedagogical software tool should be divided into independent, smaller units (modules). This facilitates the management of large systems, ensures that modifications in one module do not affect others, and enhances the potential for module reuse.

2.Reusability Principle. Previously developed components can be reused in other projects, saving time and resources. Additionally, this reduces the likelihood of errors, as the components have already been tested.

3.Minimalism Principle. Pedagogical software should be free from excessive or overly complex functionality, providing only the essential features. This simplifies usage and promotes efficient resource utilization.

4.Openness and Extensibility Principle. Pedagogical software should be open and flexible to allow the addition of new functionalities. It should also be adaptable to user requirements, ensuring long-term usability.

5.Adaptability Principle. The system must operate under diverse conditions and adjust to varying needs, providing convenience for different users. The software should also be compatible across multiple platforms.

6.Fault Tolerance Principle. The software should continue to operate even in the event of errors. This enhances the user experience while preventing unexpected failures from halting system operations.

7.Efficiency Principle. Pedagogical software should make effective use of resources, requiring minimal system demands, thereby allowing broader user access and improving operational performance.

Adherence to these design principles enhances the quality, effectiveness, and user experience of pedagogical software tools. They ensure system stability, adaptability, and security, transforming the software into a reliable, long-term educational resource.

Based on the outlined requirements and principles, it can be argued that enhancing students' competencies in the design of pedagogical software tools necessitates the creation of an open information-educational environment on the Internet. This requires adherence to principles developed by researchers for the design of such information-educational environments. Accordingly, within this study, the principles and requirements proposed by U.N. Taylakov, S.Q. Tursunov, Y.A. Vinnitskiy, V.P. Demkin, U. Lidwell, K. Holden, D. Butler, S. Wensheyk, and V.A. Krasilnikova concerning information-educational environments and electronic learning resources were analyzed.

From this analysis, we conclude that the development of an information-educational environment to support students' competencies in pedagogical software design should be guided by the following principles:

1.Design Principle of the Information-Educational Environment. The environment created for the course "Application of Information Technologies in Professional Activities" should feature visually appealing colors, legible fonts, and simple animation effects to enhance engagement.

2.User-Centered Principle. The primary focus in designing the information-educational environment should be on the users (students and instructors), taking into account their needs, abilities, and learning styles.

3.Interactivity Principle. This principle ensures interactive teaching and feedback through specialized tools. It involves processing student actions via computer-based learning systems, responding to the activities of other participants (instructors and students), facilitating direct instructor participation, and providing results for joint discussion among the learning process participants.

4.Modularity Principle. Components of the information-educational environment should be independent and organized into separate modules, enabling students to study autonomously. Different aspects of programming, such as syntax, algorithms, and data structures, should be structured into distinct modules.

5.Interactive and Collaborative Principle. Pedagogical programming instruction should employ active (learning-by-doing) and collaborative approaches. Students are encouraged to exchange ideas and work together, including engaging in group projects and tasks that foster experience sharing.

6.Automated Assessment and Analysis Principle. Automated tools should be implemented within the information-educational environment to evaluate students and analyze their learning processes, supporting effective learning management.

7.Cross-Platform Principle. The information-educational environment should be accessible across multiple platforms, allowing students to participate through various devices (computers, tablets, smartphones), thereby enhancing accessibility and flexibility.

References:

1.R.U. Umarova. (2022). Utilization of pedagogical technologies in the educational process. Economics and Society, 3(94-2). Retrieved from www.iupr.ru

2.B.B. Hamidov & D.O. Kamolova. (2022). The significance of pedagogical software tools in teaching technological education. Vestnik Magistratury, 4-1(127). ISSN 2223-4047

3.F.E. Sattorova. (2022). The importance of preparing educational materials in the learning process. Oriental Renaissance: Innovative, Educational, Natural and Social Sciences, E-ISSN 2181-1784. Retrieved from www.oriens.uz

4.I.R. Yacubova. (2021). Developing professional competencies of Turkic-speaking students in learning Russian as a foreign language (based on the author's interactive program "Russian for Military Lawyers"). In II International Congress "Language Policy of the CIS Countries" (pp. 207–209).

5.M.R. Muratovich & Q.A. Abdurahmonovich. (2021). Children's and girls' community learning and raising their children's community. Academicia Globe, 2(10), 92–98.