A Development System for Model-Training Tutors

John R. Anderson and Ray Reeder

Carnegie Mellon University, Pittsburgh, PA 15237 USA
Department of Psychology

Abstract: Our model-training methodology for turtuins is based on the ACT-L theory.
A Development System for Model-Training Tutors

A development system for model-training tutors (Anderson & Pearson, 1989) is an important tool that allows educators and instructional designers to create and use tutors that are knowledge-based and can adapt to individual learners. The system provides a framework for developing tutors that can be tailored to different subjects and learning styles.

The system is based on the idea of a model of knowledge that represents the learner's understanding of a particular domain. The model is updated as the learner interacts with the tutor, and the tutor adapts its behavior based on the learner's progress.

One of the key components of the system is the model of student behavior (MSB) which represents the learner's cognitive processes and strategies. The MSB is used to predict the learner's behavior and to adjust the tutor's behavior accordingly.

The system also includes a knowledge base (KB) that contains information about the domain being taught. The KB is used by the tutor to generate questions and feedback that are appropriate for the learner's level of understanding.

The development system provides tools for creating and testing tutors, as well as for evaluating their effectiveness. It also includes a comprehensive set of documentation and examples to help educators get started with using the system.

Overall, the development system for model-training tutors is a powerful tool that can be used to create effective and adaptive learning environments.
The Production Process, and Model Tendency...
The importance of the production process cannot be overstated in the context of software development. The quality of the product directly impacts the satisfaction of the end-users. Therefore, it is crucial for developers to follow established best practices and methodologies to ensure the success of the project. This involves understanding the requirements, designing a robust architecture, implementing the software, and testing it thoroughly.

Summary

Programming productivity is a measure of the productivity of programmers in terms of their work performance. The productivity of programmers is influenced by various factors, including their experience, proficiency in the programming language, and the complexity of the project. To improve productivity, developers should focus on optimizing their work processes and utilizing tools and technologies that enhance their efficiency.

Defining Instruction

Knowledge Management and Mastery Instruction

The best programmers are those who have mastered the art of programming. These individuals can not only write code but also understand the underlying principles and concepts. They have a deep understanding of the programming language and can apply it effectively to solve complex problems. This mastery is achieved through continuous learning and practice.

Anderson & Polacek
experience of curriculum development. It is the component where most of the time is spent and it serves to organize subsequent development such as the design of declarative instruction. Interface design is separate but is largely a one-time initial investment whereas each unit of curriculum requires a fresh, full effort at production-rule modeling. Our system has made it easier to develop these rules and once developed provides an overall system in which they can organize efficient instruction. However, we see no way of relieving the developer of the essential question in our approach which is "How should the student think about solving this class of problems?"

This research was supported by Contract MDA-903-89K-0190 from the Army Research Institute and Grant 87-51890 from the National Science Foundation.

References


