УДК 621.396.41

Ю.А. Богомолов, А.А. Бодрухин

Облачная информационная система обучения студентов

Обучение – процесс трудоёмкий. Но ведь мы живём в эру информационных технологий: зачем всё делать самому, если можно переложить часть работы на компьютер? В рамках данной статьи будет предложен концепт системы, которая позволит упростить и улучшить многие аспекты обучения.
Ключевые слова: система, автоматизация, обучение, упрощение, студент, преподаватель.

Описание проблемы

Несмотря на все достижения научно-технического прогресса, в современном образовании мало вещей являются автоматизированными. Проверка домашних заданий в школе, лабораторных работ в университете – всё это целиком ложится на плечи учителей и преподавателей. Одни и те же задания по математике, физике, химии – решаются многими поколениями школьников и студентов, позволяя им списывать. Большая часть упражнений до сих пор выполняется в тетрадях, хотя практически у каждого уже есть компьютер, а многие печатают в разы быстрее, чем пишут ручкой. Вследствие всего этого заинтересовать учащихся учёбой становится всё сложнее.

Предлагаемое решение

Решением проблемы может стать система, которая сможет автоматизировать процессы проверки решений и генерации уникальных задач; позволит создавать учебный материал нового поколения, отличающийся новизной, интерактивностью и разнообразием; позволит учащимся выполнять задания на компьютерах и мгновенно получать отклик, что может их больше заинтересовать.

Описание концепта

Предлагаемая концепция выражается следующими требованиями к системе:

1. должна содержать множество инструментов для разработки учебных материалов, а также поддерживать их добавление;
2. преподаватели могут выдавать индивидуальные или коллективные задания;
3. учащиеся могут выполнять задания поодиночке или в группах;
4. перед тем как решения попадут к преподавателю, они пройдут автоматизированную проверку.

Для выполнения этих требований предполагается использовать следующие роли пользователей: автор, преподаватель, разработчик и студент. У каждой из ролей есть свои функции:

1. автор составляет учебный материал;
2. преподаватель выдаёт задания учащимся, а также проверяет результаты их работы;
3. разработчик создаёт программное обеспечение, дающее простор автору в составлении материала;
4. студент выполняет задания, выданные преподавателем.

При этом один и тот же пользователь может иметь несколько ролей. Приведём простой пример: аспирант и учится сам, и может быть ассистентом преподавателя, проводя лабораторные работы у студентов.

Следующей важной частью данной концепции являются продукты работы разработчика: конструкторы, интерпретаторы и чекеры.

1. Конструкторы позволяют автору создавать учебный материал, в том числе интерактивный.
2. Интерпретаторы отображают материал на странице и взаимодействуют с пользователем.
3. Чекеры проверяют решения студентов, если материал предполагает ответ.

Примеры реализации взаимодействия пользователей и системы

Теперь пойдём по примерам схем, отображающим взаимодействие интерпретаторов, чекеров и пользователей. На рисунках 1 и 2 изображены простейшие схемы типов «Тест» и «Эссе по литературе». В первом случае мы видим тривиальную схему, аналогичную работе любого преподавателя. Работа достаточно механическая, потому легко упрощается предлагаемой системой.

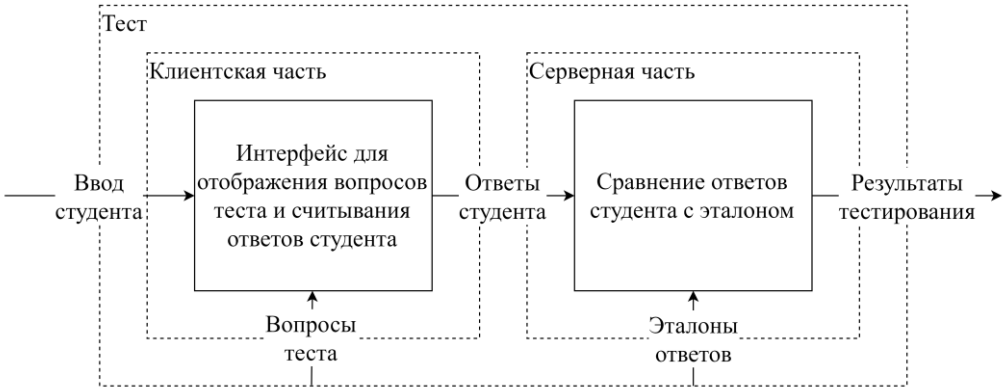

Рисунок 1. Тест

Хотя схематически рисунки 1 и 2 идентичны, следует уточнить, что «Данные о других эссе» могут не задаваться напрямую преподавателем, а, например, автоматически извлекаться из архивов готовых сочинений. Мониторить сайты с готовыми ответами уже не входит в обязанности преподавателей, поэтому студентам легко схитрить и списать. Результаты автоматической проверки на плагиат позволят преподавателям усомниться в добросовестности студентов, и ставить более заслуженные отметки.

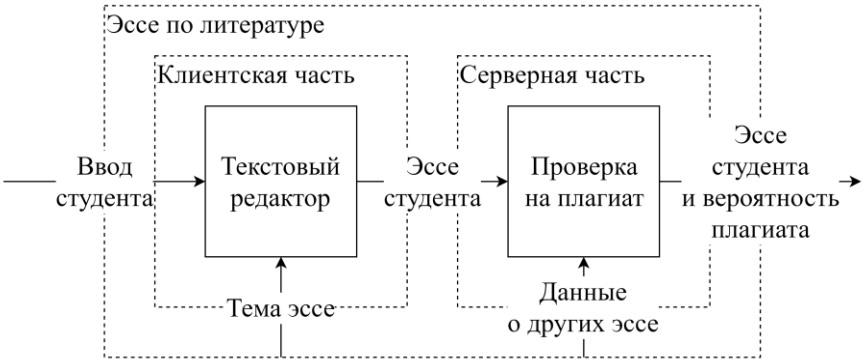

Рисунок 2. Эссе по литературе

На рисунке 3 можно увидеть более сложную схему – «Задача спортивного программирования». Как правило, именно такая схема используется на любых олимпиадах по программированию, а также тренировочных веб-ресурсах, вроде Codeforces. С помощью этой же схемы будет можно проверять и обычные лабораторные работы студентов по программированию.

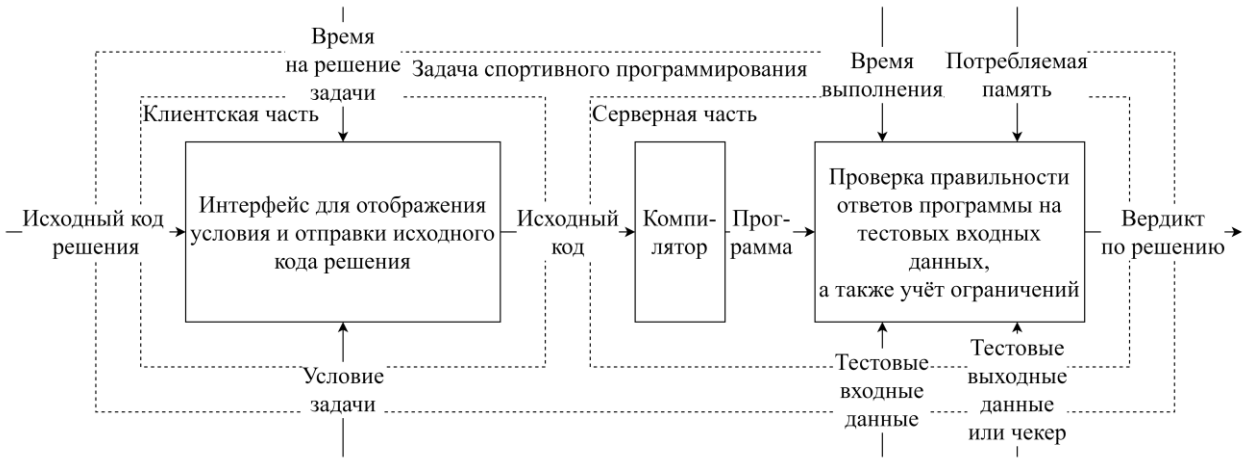

Рисунок 3. Задача спортивного программирования

На рисунке 4 можно увидеть не менее сложную схему – «Задача по математике». Как и в случае с тестом, работа преподавателя здесь достаточно механическая, но в то же время на порядки сложнее, чем просто сравнить два столбца символов. Проверка сотни работ с десятками интегральных задач в каждой может занять неделю, а то и больше. Автоматическая проверка значительно снизит нагрузку на преподавателя, позволяя ему сосредоточиться на процессе обучения, а не проверки.

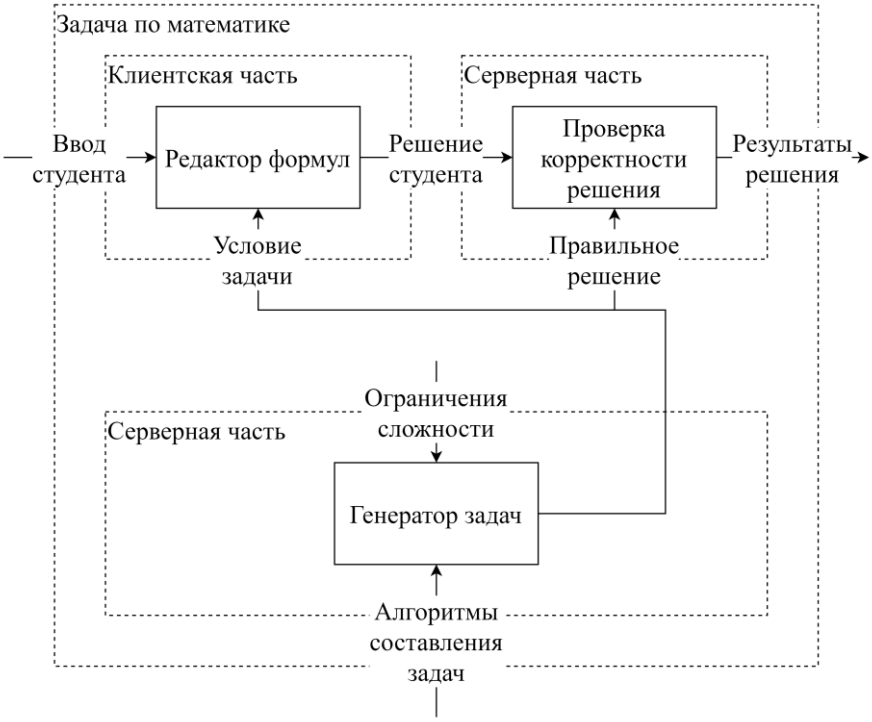

Рисунок 4. Задача по математике