НАУЧНОЕ ИЗДАНИЕ МГТУ ИМ. Н. Э. БАУМАНА
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС77 - 48211. Государственная регистрация №0421200025. ISSN 1994-0408
электронный научно-технический журнал

Web-ориентированные интеллектуальные обучающие системы на основе нечеткого деятельностного подхода в обучении

 # 11, ноябрь 2012
DOI: 10.7463/1112.0489620

Клюкин В. Э.

УДК 007.52

Россия, г. Екатеринбург, Уральский федеральный университет. Физико-технологический институт
vt@dpt.ustu.ru
kve2310@gmail.com

Introduction

Компьютерное обучение – интересная и перспективная область исследований, привлекающая передовых ученых, педагогов всего мира. Интенсивно исследуются различные пути добавления **адаптивной** и **интеллектуальной** функциональности обучающим системам, чтобы сделать их привлекательными для каждого студента. Именно web-ориентированные адаптивные интеллектуальные обучающие системы (ИОС) являются сейчас основными объектами исследований и разработок, [1, 2].

Одной из наиболее прогрессивных многообещающих концепций в образовании является *деятельностный подход*, при котором традиционное обучение, как усвоение «готовых» знаний, заменяется усвоением знаний по результатам *деятельности* обучаемых, формирующей их содержание и качественные характеристики [3]. Деятельностный подход в обучении опирается на индивидуальные характеристики и предпочтения обучаемого, поэтому он особенно перспективен для web-ориентированных обучающих систем, предназначенных для дистанционного обучения и самообразования.

В настоящее время разработано много различных стратегий обучения и моделей ИОС, включая и обучающие системы с деятельностным подходом. Большинство исследований по интеллектуальным обучающим системам посвящено моделированию дидактики и выполнено на громоздком универсальном инструментарии систем искусственного интеллекта. Разработки же ИОС для практического обучения, особенно web-ориентированные и масштабируемые, исчисляются единицами, [4, 5]. Ощущается разрыв между специалистами по дидактике и разработчиками такого программного обеспечения. Мало работ, где перспективные дидактические подходы разъяснялись бы энтузиастам-разработчикам ПО, где обсуждались бы ***web*-ориентированные масштабируемые**

архитектуры обучающих систем. Ведь ясно, что далеко не все из хорошо зарекомендовавших себя в локальном исполнении подходов пригодны для перевода их в Интернет.

Одна из целей настоящей работы – инициировать обсуждение *между дидактиками и разработчиками ПО* технологий разработки и перспективных архитектур компьютерных обучающих систем.

Для начала обсуждается архитектура обучающей системы на основе модифицированного деятельностного подхода в обучении, названного автором *Нечетким деятельностным подходом* (the fuzzy activity approach in teaching). *Нечеткий деятельностный подход* представляет собой известный в дидактике деятельностный подход, в котором модель обучаемого и процесс обучения основаны на нечетких знаниях и методах искусственного интеллекта. Предлагается *двухслойная многопотоковая архитектура ИОС*, позвлояющая разделить функциональную ответственность программных модулей, повысить устойчивость системы к изменению дидактической модели процесса обучения, обеспечивающая многоразовый обмен информацией между клиентом и сервером в пределах одной сессии. В работе обсуждается:

- ИОС на основе Нечеткого деятельностного подхода в обучении;
- двухслойная многопотоковая организация процесса обучения в ИОС;
- использование нечеткой логики и нечетких множеств, позволяющее ввести мощный унифицированный формат описания знаний на основе логики предикатов ППП, дополненный описанием нечетких знаний;
- использование языка Пролог в логических подсистемах ИОС, технология встраивания языка SWI-Prolog в PHP;
- оригинальная клиент/серверная архитектура ИОС с решением проблемы обратных вызовов клиента;
- Унифицированный технологический процесс разработки компьютерных обучающих систем.

1. Нечеткий деятельностный подход в обучении

1.1. Деятельностный подход и нечеткие знания

Деятельностный подход в обучении, развиваемый Г. А. Атановым [3], очень близок автору, преподающему искусственный интеллект на физтехе, который всегда был сторонником «изучения на примерах». Думается, что рассматриваемую Г. А. Атановым связку «дидактика и искусственный интеллект» можно расширить использованием нечетких знаний со стороны искусственного интеллекта в обучении. Это позволило бы создавать более эффективные сетевые ИОС на основе представления и использования знаний в процессе обучения и в модели обучаемого, как нечетких.

1.1.1. Деятельностный подход в обучении

Использование искусственного интеллекта в обучении – новая методология психологических и дидактических исследований, ориентированная на моделирование

поведения человека в процессе обучения. Объяснить сущность деятельностного подхода, можно ответив на вопрос: «Зачем учить человека?». Ответ очевиден: затем, чтобы он умел потом действовать, эффективно работать по специальности. Таким образом, *цель обучения* – научить *способам выполнения действий* при решении задач, умению использовать подходящие знания, чтобы *выполнить действие*. Это положение принято за фундамент, на котором строится деятельностный подход в обучении. Знания выступают как *средство* для эффективного выполнения действий. Нельзя вначале приобрести знания, а затем приступить к их использованию. Во-первых, это будет запоминанием, а не усваиванием знаний. Во-вторых, приобретение знаний можно превратить в нескончаемую самоцель, не генерируя ничего полезного. Истинное приобретение знаний возможно только через процесс выполнения какой-нибудь работы с их помощью. Не используемые в деятельности знания можно считать балластом. Фундаментом такой методики обучения является *деятельность усвоения* как механизм всех приобретений учащегося, планируемых целями обучения, [6]. Такова кратко суть деятельностного подхода в обучении.

1.1.2. Нечеткий деятельностный подход

Подход к обучению, названный автором *нечетким деятельностным подходом*, представляет собой модификацию деятельностного подхода, в которой модель обучаемого и процесс обучения рассматриваются, как основанные на нечетких знаниях и методах искусственного интеллекта для обработки нечетких знаний. Использование нечеткой логики и нечетких множеств позволяет ввести мощный унифицированный формат описания знаний на основе логики предикатов ППП, дополненный описанием нечетких знаний. А это приводит к более совершенным моделям обучаемого и процессам обучения, улучшенным механизмам адаптации и более эффективным компьютерным обучающим системам, особенно web-ориентированным и масштабируемым. Подход открыт как для новых дидактических исследований, так и исследований по нечетким знаниям в искусственном интеллекте. Рассмотрим связку «дидактика и искусственный интеллект» со стороны искусственного интеллекта.

1.1.3. Представление и использование нечетких знаний в обучении

Успешное обучение невозможно без широкого применения современных методов извлечения, обработки и систематизации знаний. Моделирование умственных способностей человека и методы работы со знаниями являются предметом искусственного интеллекта, поэтому рассмотреть поведение человека в процессе обучения с позиций искусственного интеллекта, как *средства* достижения учебных целей, более чем оправдано.

В процессе обучения преподаватель передает информацию обучаемому, используя разнообразные средства передачи информации: голос, интонацию, жесты, рисунки. Основным средством передачи информации в диалоге обучающего и обучаемого при этом служит *естественный язык* (ЕЯ). Однако, с точки зрения обмена информацией естественный язык имеет четыре существенных недостатка: он неточен, неоднозначен, неполон и избыточен, [7]. Совершенно очевидно, что бо́льшая часть информации в диалоге не выражается определенно и ясно. Когда оба собеседника одинаково хорошо знают

проблему, то это приводит к очень сжатым и четким сообщениям. Недаром говорят: «Мудрому достаточно услышать от мудрого одно слово». Но поскольку при обучении обоюдного знания предмета у преподавателя и у студента нет, то часто возникает ошибочная интерпретация сказанного.

Понимание произнесенной фразы и степень образованности слушающего – две стороны чрезвычайно важной проблемы кодирования и декодирования в системах с ЕЯ-сообщениями. Отсюда, при взаимодействии учителя и ученика возникает проблема *передачи обучаемому нечетких знаний*: процесс обучения постоянно ставит перед обучаемым *неформализованные задачи*. Фундаментальной отличительной особенностью таких задач является недетерминизм, свобода действий. В них всегда предполагается наличие выбора между многими вариантами в условиях неопределенности. Такие задачи являются трудно решаемыми: их решение, чаще всего, приходится искать методом проб и ошибок, причем число таких проб очень велико.

Искусственный интеллект является наиболее эффективным направлением решения неформализованных задач обработки нечетких знаний, решая их в диалоге с ЭВМ, [8]. Хотя диалог и создает определенные технологические трудности для web-ориентированных обучающих систем, основная проблема решения задач обучения связана с наличием *нечеткости, с нечеткими знаниями*.

1.1.4. Вероятность и нечеткость

В связи с представлением и использованием нечетких знаний, несколько слов о понимании нечеткого и вероятностного рассмотрения. Вероятность – характеристика явлений, имеющих массовую статистическую природу. Со стороны математики вероятностные модели опираются на математическую статистику, статистическую теорию решений, критерии среднего риска, вероятностные оценки, [9]. *Нечеткость* – характеристика явлений другой природы, недостаточно изученных, информация о которых неполна, неоднозначна, а иногда и ошибочна. Со стороны математики нечеткие модели опираются преимущественно на прикладную теорию нечетких множеств, на соответствующие многокритериальные модели принятия решений, [10]. Основное достоинство нечетких множеств, как средства описания моделей принятия решений в условиях неопределенности в том, что они позволяют учесть в выводе одновременно все возможные варианты (логические посылки), каждый со своим весом. Это давно поняли специалисты по искусственному интеллекту и плодотворно используют нечеткие множества в самых различных моделях. Методы обработки нечетких знаний в обучающих системах продемонстрируем при рассмотрении развиваемого автором *нечеткого деятельностного подхода в обучении*.

1.2. Модель учебного процесса

Основное назначение ИОС – предоставлять пользователю знания в виде *адаптивных* учебных ресурсов и *научить использованию* этих знаний для решения задач. Главная особенность таких ресурсов – возможность *адаптации качества учебного материала*, зависящей от уровня подготовки обучаемого и его предпочтений, и *последовательности*

обучения, зависящей от конкретных целей пользователя. Поэтому в ИОС должны присутствовать две главные составляющие – это *база знаний* предметной области курса и *механизм моделирования обучаемого*. Центральная идея разработки адаптивных компьютерных систем обучения состоит в оптимизации процесса изучения курса путем подстройки обучающей среды под особенности конкретного ученика. Для этого необходимо:

- построить эталонную модель курса;
- создать адаптивную модель обучаемого;
- построить модель учебного процесса как совокупность стратегий обучения, оперирующих обучающими воздействиями, оптимальными для той или иной модели обучаемого;
- разработать подсистему контроля деятельности обучаемого и генерации управляющих решений для достижения конечной цели обучения.

Рассмотрим основные из этих составляющих.

1.3. Эталонная модель курса

Информационным фундаментом *эталонной модели курса* (ЭМК) служат объекты предметной области и отношения между ними. Модель ПО удобно представить в виде тезауруса, дескрипторные группы которого описывают родо-видовые отношения между объектами ПО. Дополнительно тезаурус решает и вопросы терминологии, выделяя дескрипторы, как основные термины будущего курса, и вспомогательные именования понятий (синонимы, акронимы, псевдонимы). Еще лучше выстроить описание ПО в виде композиции ГЛОТЕОН (глоссарий, тезаурус, онтология), что позволило бы использовать нечеткие логико-лингвистические методы к выстраиванию процесса обучения, [12]. В данной работе эти методы не обсуждаются.

Перейдем непосредственно к ЭМК и построим структурную модель курса. Введем конечное множество $X = \{x_i\},\ i = \overline{1,N}$, где x_i – обучающий блок (соответствует порции учебного материала). Для выстраивания последовательности изучения курса введем *отношение связности по информации* α между обучающими блоками, а для выстраивания иерархии тем изучаемого материала – *отношение детализации* β:

$$x = \alpha(x_1, x_2, \dots, x_N), \tag{1}$$

$$x = \beta(x_1, x_2, \dots, x_n). \tag{2}$$

Формулы (1) и (2) выражают то, что каждое знание x состоит из знаний $X = \{x_i\}$, а кортеж $\langle X, \alpha, \beta \rangle$ описывает всю структурную модель курса.

Если ввести обозначения A_i как входные знания, передаваемые от каждого из обучающих блоков x_i к обучающему блоку x, а B, как единственное выходное знание блока x, то справедлива формула вывода в виде импликации:

$$x(A_1, A_2, \dots, A_i, \dots, A_n) \to B. \tag{3}$$

Сеть потоков вывода с импликациями вида (3) составляет ядро базы знаний рассматриваемой ИОС с универсальным форматом правил вывода, который можно записать (например, на языке Пролог) в виде предиката-импликации `imp(), [11]`:

```
imp(T,B,As,Ss,C),                                    (4)
```

где T – тип объединения потоков знаний (И, ИЛИ);

B – идентификационный номер выходного знания (заключения);

As – список номеров входных знаний (посылок);

Ss – список знаков посылок;

C – оценка уверенности эксперта в данной импликации (число в интервале [0, 1]), учитываемая решателем задач при обработке нечетких знаний.

Для перехода к нечеткой модели процесса обучения представим эталонную модель курса нечетким множеством $\tilde{S}_0$ – совокупностью пар вида

$$\tilde{S}_0 = \{< \mu_{S_0}(i)/i >\}, \quad i \in I, \ \mu_{S_0}(i) \in [0,1]\,, \tag{5}$$

где I – базовое множество тем курса (*концептов*);

$\mu_{S_0}(i): I \to [0,1]$ – функция принадлежности i–го концепта к эталонной системе знаний курса;

i – номер концепта (идентификатор его адреса в базе знаний ИОС).

Значения функция принадлежности $\mu_{S_0}(i)$ назначаются преподавателем. Например, чтобы поставить перед обучаемым задачу – овладеть i–м концептом в совершенстве, он назначает $\mu_{S_0}(i) = 1$. Если обучаемый сам определяет цель изучения курса, то он сам и задает эти значения (естественно автоматизированным способом). Таким образом, значения функции принадлежности μ_{S_0} в эталонной модели курса определяют цель обучения – заданную степень овладения навыками/умениями: ($\mu_{S_0}(i) = 0$ – совсем не знает, $\mu_{S_0}(i) = 1$ – владеет в совершенстве). Рассмотрим адаптивную модель обучаемого при нечетком деятельностном подходе.

1.4. Адаптивная модель обучаемого

Модель обучаемого является одним из основных понятий современной дидактики. Это понятие возникло в компьютерных технологиях обучения в связи с необходимостью формализовать представления об обучаемом, [13, 14]. В настоящее время нет установившегося определения – что понимать под термином «Модель обучаемого». В самом общем виде *модель обучаемого* – это знания о нем, используемые для организации процесса обучения [3]. Это множество фактов о состоянии обучаемого и, прежде всего, о его предметных знаниях и умениях. Модель состоит из набора характеристик обучаемого, измеряемых во время обучения и определяющих степень усвоения знаний. Моделирование обучаемого является развивающимся направлением искусственного интеллекта в обучении. Модель обучаемого лежит в основе адаптивных механизмов настройки процесса обучения. Основная информация, которая должна присутствовать в модели обучаемого:

- цель обучения;
- знания обучаемого в рамках изучаемого курса;
- особенности подачи учебных материалов и вопросов по их проверке;
- способы предоставления контрольных заданий по проверке навыков/умений;
- правила модификации модели обучаемого по результатам его обучения.

1.4.1. Классификация моделей обучаемого

Общепринятая классификация делит модели обучаемого на две основные группы: *фиксирующие* и *имитационные*. Фиксирующие подразделяются на оверлейные (векторные и сетевые) и генетические графы, а к имитационным относят в основном модели ограничений, ошибок и фальшправил. По способу представления знаний все существующие модели подразделяются на декларативные, процедурные и распределенные, [15, 16]. Так к декларативным моделям относят *скалярную* («знает»/«не знает» или n-бальная оценка; это – грубая модель), *стереотипную* («отличник», «хорошист», «двоечник»; это – простая модель) и *оверлейную* (элементарные единицы знаний – *концепты* с параметрами, характеризующими степень их усвоения обучаемым; это – мощная и гибкая модель). Хотя каждый вид модели имеет свое предназначение, области их действия в значительной степени пересекаются. Поэтому на практике чаще используют комбинации моделей или их модификации.

Оверлейная модель строится в предположении, что знания обучаемого и знания системы имеют одинаковую структуру, при этом знания обучаемого являются подмножеством знаний системы. С каждой темой предметной области связывается числовой атрибут, показывающий степень правильного понимания обучаемым материала по этой теме. Значение этого атрибута определяется в ходе тестирования обучаемого. Оверлейная модель хорошо распознает ошибки обучаемого как его пробелы в знаниях по сравнению со знаниями эксперта. Если же ученик применил оригинальный подход, отличный от экспертного – оверлейная модель бессильна, [17]. Исправить ситуацию можно введением сети процедур поведения обучаемого при выполнении заданий, вплоть до его примитивных действий. Для этого разработаны более сложные модели: *разностная* и *пертурбационная*.

Разностная модель использует сравнение ответов обучаемого с экспертными знаниями системы для обнаружения у него альтернативных подходов к решению задач. Она не только поправляет неправильное использование имеющихся знаний при обучении навыкам/умениям, но и позволяет ближе подойти к оценке творчества ученика. Структура знаний обучаемого и системы при этом продолжают предполагаться идентичными.

Наиболее сложна *пертурбационная модель*. Она вообще не накладывает ограничений на полную идентичность структуры знаний пользователя и системы. Модель строится в предположении, что знания обучаемого и знания системы могут частично не совпадать. В этом случае важной предпосылкой построения такой модели является идентификация *причин расхождения*, т.к. без определения причин расхождений модель обучаемого становится слишком неопределенной.

Нечеткий деятельностный подход проявляется в любой модели обучаемого. Данная работа не предназначена для систематизации моделей обучаемого, поэтому рассмотрим нечеткий подход только на примере популярной *оверлейной модели*. Рассмотрение начнем с предметной модели обучаемого, определяемой эталонной моделью курса и моделью предметной области.

1.4.2. Обучаемый как объект управления

Обучаемый может рассматриваться как сложный объект управления, а процесс обучения – как процесс управления этим объектом, [18]. Такой подход позволяет, используя общую теорию управления, проанализировать зависимость качества и поставленных целей обучения от характеристик индивида, конкретной личности с ее индивидуальными свойствами. На рис. 1 показана модифицированная модель Растригина, отображающая процесс управления обучаемым. Модификация связана с интеллектуальным подходом к обработке нечетких знаний во время учебного процесса.

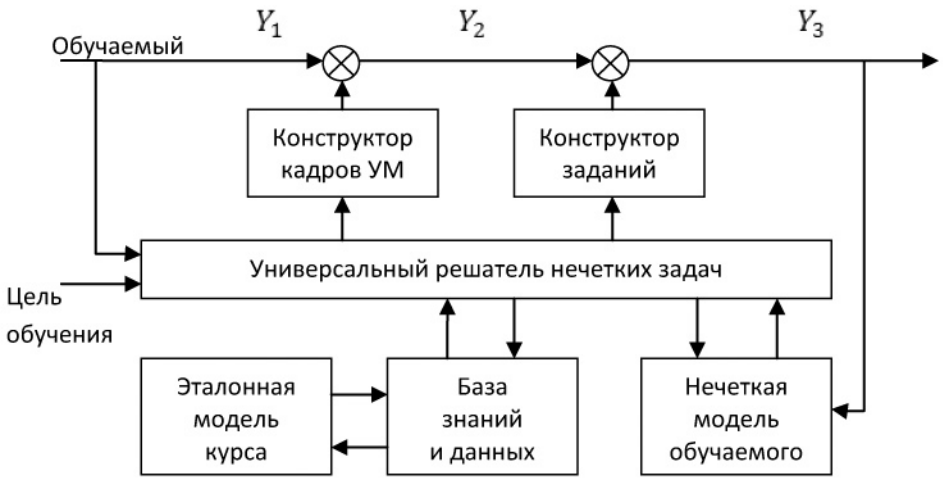

Рис. 1. Модель учебного процесса при нечетком подходе.

В начале акта управления (обучения) обучаемый находится в состоянии Y_1. На основе информации, хранящейся в базе знаний и данных (БЗнД), а также модели обучаемого, с использованием алгоритмов обработки нечетких знаний формируются кадры учебного материала (обучающие воздействия). В ходе изучения предоставленной информации обучаемый получает новые знания и переходит в состояние Y_2. Для проверки навыков/умений управляющая система вырабатывает тестовые вопросы и задания, выполняя которые обучаемый переходит в состояние Y_3.

База знаний и данных формируется на основе эталонной модели курса. Ее назначение – преобразовать формат эталонной модели курса в согласованный с универсальным решателем

нечетких задач (УРНЗ). Блоки БЗнД и УРНЗ составляют ядро интеллектуальной системы. В базе данных хранится статическая информация об эталонной модели курса, об обучаемых и др. База знаний содержит набор правил для решателя, находящего решения неформализованных задач обучения для поставленной цели обучения.

Какова цель обучения, чему необходимо научить студента? Как должна быть устроена система знаний в ИОС для эффективного обучения? Как построить последовательность обучения (движение обучаемого от исходных знаний к целевым)? Как контролировать усвоение знаний? Как оценивать навыки/умения, приобретаемые обучаемым? Ответ на эти вопросы начнем с построения модели обучаемого, как объекта управления.

1.4.3. Нечеткая модель обучаемого

Технологически модель обучаемого – это способ представления и хранения совокупности личных характеристик пользователя ИОС. Знания о том, каким мы хотим видеть обучаемого, требования к его конечному состоянию (как к специалисту) составляют *нормативную* модель обучаемого. Часть нормативной модели, определяющую предметные знания, называют *предметной моделью обучаемого*. Именно эту модель мы и будем рассматривать далее. Задача – адаптировать систему к потребностям ученика. Для этого модель обучаемого должна включать информацию: 1) о конечной цели обучения, 2) о текущих результатах учебной деятельности, 3) об особенности изложения учебных материалов и выбора контрольных заданий и вопросов, 4) о правилах модификации модели обучаемого по результатам его работы.

Модели обучаемого могут иметь разное строение: одни являются хранилищем знаний об обучаемом, другие представляют собой последовательность его действий. В настоящей работе рассмотрена web-ориентированная ИОС с нечеткой комбинированной адаптивной моделью обучаемого, формируемой в процессе обучения. Модель интерактивно динамически корректируется по результатам обработки нечетких знаний. Предложенная модель обучаемого описывается с помощью нечетких множеств, учитывает явно текущие знания и умения индивида, а косвенно и его психологические характеристики – запоминание, забывание и предпочтения. Формат знаний о свойствах обучаемого в виде нечетких множеств достаточно универсален. Это дает возможность написать для сетевой ИОС программу универсального решателя нечетких задач, которая позволяет интерактивно управлять свойствами и даже структурой модели обучаемого с учетом накопленного опыта, гибко корректировать их без перепрограммирования системы в целом.

По данным [19] можно выделить следующие характеристики обучаемого, которые надо учитывать в параметрах модели:

1. уровень знаний;
2. психологические характеристики (тип личности, ориентация и др.);
3. стиль изучения материала;
4. скорость усвоения;
5. уровень умений и навыков при выполнении заданий;
6. внимание и способность к обучению (очень внимательный, средне, мало).

Теория построения модели обучаемого в настоящее время еще остается нерешенной проблемой дидактики, хотя целый ряд практических моделей обучаемого уже построено и успешно используются в различных компьютерных системах, [20]. Как было отмечено, большинство исследователей подразделяют модели обучаемого на два основных класса: *скалярная* модель с интегральной оценкой знаний обучаемого, например по пятибалльной шкале, и *оверлейная* модель с селективной оценкой знаний (что именно знает обучаемый и чего не знает), опирающаяся на предметную область курса. Интеллектуальные обучающие системы обычно используют оверлейную модель, т. к. для организации эффективного процесса обучения учителю на каждом шаге обучения требуются детальные сведения о текущих знаниях обучаемого в сравнении с эталонными.

Представляется разумным использовать следующую комбинацию моделей. В начале процесса обучения, пока детальных данных о пользователе еще недостаточно для построения адекватной адаптивной модели, можно применить простую *стереотипную* модель обучаемого, отражающую его индивидуальность в интегрированном виде: «отличник», «хорошист», «двоечник». А по мере накопления информации переходить к *оверлейной* модели. Формально стереотипную модель можно описать следующим образом.

Пусть система содержит N стереотипов, которые обозначим U_1, U_2, …, U_N. Принадлежность к стереотипу определим с помощью M неравнозначных критериев. Для критериев введем обозначения C_1, C_2, ,…, C_M. С каждым критерием свяжем нечеткое множество, и принадлежность к стереотипу определим с помощью оценочной функции (функции принадлежности этого нечеткого множества) $\mu_{C_i} \in [0,1]$, $i = \overline{1,M}$. Тогда результирующий критерий D_k стереотипа U_k можно рассчитать по формуле

$$D_k = C_1^{\beta_1} \& C_2^{\beta_2} \& \dots \& C_M^{\beta_M}$$

где β_i учитывает неравнозначность критериев C_i. Величины β_i можно вычислить методом парных сравнений критериев. Пусть матрица парных сравнений критериев C_i по n-бальной шкале равна

$$\overleftrightarrow{A} = \begin{bmatrix} 1 & \cdots & a_{1M} \\ \vdots & \ddots & \vdots \\ a_{M1} & \cdots & 1 \end{bmatrix}$$

Находим максимальное собственное значение c_M матрицы $\overleftrightarrow{A}$, затем находим собственные вектора для этого собственного значения $\omega_j(c_M)$, $j = \overline{1,M}$ для c_M. Нормируя полученные значения на ранг матрицы $\overleftrightarrow{A}$, получаем окончательно

$$\beta_i = M\omega_i(c_M) \qquad (6)$$

Собственные значения и собственные вектора матрицы можно вычислить в MATHCAD с помощью имеющихся там функций *enginval()* и *enginvec()*.

Когда информации о пользователе становится достаточно для анализа и последующей адаптации, переходим на *оверлейную* модель обучаемого. Для построения оверлейной

модели обучаемого перейдем к используемой там терминологии: x_i-блоки курса, введенные выше при описании эталонной модели курса, будем рассматривать как *концепты* оверлейной модели, структурированные иерархически. С точки зрения выводимости отметим, что согласно свойствам отношений информационной связности, концепты поделены на категории: x_t – начальные (терминальные), x_q – целевые и промежуточные (выводимые), $x_i,\ i \neq t,\ i \neq q$.

Представим концепты курса в виде кортежа $\langle j, x_j, L_j, B_j, r_j \rangle$, где j – номер концепта (идентификатор адреса), x_j – наименование (дескриптор), L_j – список номеров входных концептов (поставляющих исходные знания A_i), B_j – выходное знание или умение, r_j – число возвратов повторного изучения. Тогда аспект умений обучаемого в оверлейной модели, т. е. его текущие знания и умения по темам курса, можно представить в виде нечеткого множества (множества пар «концепт – оценка»):

$$\tilde{A} = \{< \mu_A(j)/j >\},\quad j \in J,\ \mu_A(j) \in [0,1], \tag{7}$$

где J – базовое множество тем курса, $\mu_A(j): J \to [0,1]$ – функция принадлежности.

Значения функции принадлежности $\mu_A(j)$ определяют степень овладения темой. Функция принадлежности вычисляется по результатам тестирования и решения контрольных заданий по темам курса. Рассмотрим использование функции принадлежности в процессе обучения. Для этого введем понятие *матрицы релевантности*.

1.5. Матрица релевантности

В процессе обучения навыкам/умениям обучаемому выдаются задания по изучаемой теме. Обучаемый составляет алгоритмы выполнения заданий из заданного набора типовых правил $p_i \in P,\ i = \overline{1,m}$, представляющих элементарные действия над концептами $q_j \in Q,\ j = \overline{1,n}$.

Для адаптации модели обучаемого и вычисления последовательности продвижения к цели необходимо иметь связь используемых обучаемым правил с нормативными концептами. Введем нечеткое соответствие $\tilde{\Re} = (P, Q, \tilde{F})$, где P – область отправления (множество правил $\{p_i\}$), Q – область прибытия (множество концептов $\{q_j\}$), $\tilde{F}$ – нечеткий график движения (нечеткое множество над прямым произведением $P \otimes Q$ с функцией принадлежности $\mu_{\tilde{F}}(i,j)$).

Назовем *матрицей релевантности (relevance matrix)* – матрицу инциденций нечеткого соответствия действий обучаемого нормативным концептам учебного материала. Под действиями обучаемого будем понимать его реакцию на любые виды тестирования: ответы на вопросы, решение задач, выполнение заданий. Нечеткое соответствие $\tilde{\Re}$ релевантности концептов учебного материала действиям обучаемого и матрица релевантности показаны на рис. 2.

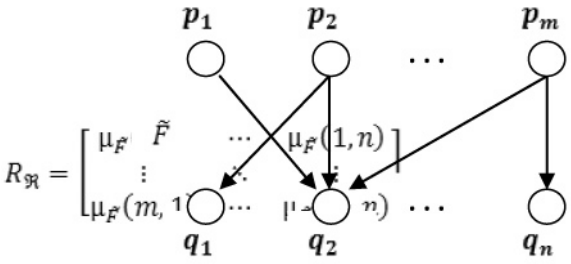

Рис. 2. Нечеткое соответствие $\tilde{\Re}$ релевантности концептов действиям обучаемого: $R_{\Re}$ – матрица релевантности

При реализации оверлейной модели умений обучаемого в виде нечеткого множества пар (5), матрица релевантности $\tilde{\Re}$ удобно связывает действия обучаемого с концептами обучающего материала, позволяя на каждом шаге обучения адаптировать модель обучаемого, оптимизировать последовательность продвижения к цели обучения. Процесс обучения заключается в сравнении на каждом «к»-м шаге обучения умений обучаемого $\tilde{A}_k = \{\langle \mu_{A_k}(j)/j \rangle\}$ с умениями, определяемыми эталонной моделью курса $\tilde{B}_k = \{\langle \mu_{B_k}(j)/j \rangle\}$. Алгоритм обучения следующий:

1. После освоения очередного блока обучения, когда обучаемый находится в состоянии Y_2 (см. рис. 1), конструктором вопросов и заданий ему дается *учебная задача*. Обучаемому выдается также набор допустимых правил (операций) $y_i : y_i \epsilon Y,\ i = \overline{1,N}$, с помощью которых он может составить свой алгоритм решения задачи. Обучаемый, выбирая из этого набора свою последовательность операций, решает учебную задачу. Операции y_i подобраны так, чтобы при неправильном их применении можно с помощью соответствия релевантности $\tilde{\Re}$ определить адреса соответствующих концептов и направить к ним обучаемого для повторного изучения. Опуская детали (они следуют из результатов [4]), приведем формулу для вычисления значений функции принадлежности $\mu_{A_k}(j)$, описывающую умения обучаемого на «к»-м шаге обучения – степень усваивания умений по «j»-му концепту курса:

$$\mu_{A_k}(j) = \frac{\sum_i r_{ij} p_i(k)}{\alpha_j} \qquad (8)$$

   Здесь r_{ij} – элементы матрицы релевантности (рис. 2), определяющие связи операций y_i с адресами концептов x_j, $p_i(k)$ – вероятность правильного применения «i»-й операции на «к»-м шаге обучения, оцениваемая приближенно(!) с использованием субъективного байесовского подхода, а $\alpha_j = \sum_i e_{ij}$ – количество операций из всего множества операций Y, в которых используется концепт x_j.

   После вычисления всех значений функции принадлежности μ_{A_k} на «к»-м шаге обучения обучаемый переходит в состояние Y_3, рис. 1.

2. Теперь сравним знания обучаемого с эталонными: т. е. вычислим степень включения $\tilde{A}$ в $\tilde{B}$:

$$v(\tilde{A},\tilde{B}) = \&_{j\in J}(\mu_A(j) \to \mu_B(j)) \quad (9)$$

где операция импликации " → " определяется согласно нечеткой логике Лукасевича как

$$\mu_A(j) \to \mu_B(j) = 1\&(1 - \mu_A(j) + \mu_B(j)) \quad (10)$$

При $v(\tilde{A},\tilde{B}) < 1$ фиксируем недостаточные умения обучаемого на «к»-м шаге обучения. Используя $\mu_{A_k}(j)$ из (6) определяем номера плохо усвоенных концептов j, а сортируя значения $\mu_{A_k}(j)$ по минимуму, находим наилучшую для обучаемого последовательность повторения неусвоенных концептов.

Таким образом, на основе реализации оверлейной модели умений обучаемого, использующей нечеткие множества и типичные в искусственном интеллекте способы обработки нечетких знаний, можно организовать адаптивное управление обучаемым, заключающееся в выдаче ему на очередной шаг обучения тестовой задачи оптимальной сложности и направленности.

1.6. Формат описания знаний

Нечеткая логика – разновидность непрерывной логики, в которой логические формулы могут принимать истинностные значения в промежутке от 1 до 0. Пусть t_x и t_y – истинностные значения посылок X и Y некоторого правила, заключение $t_{зак}$ которого получено с субъективной вероятностью $t_{правила}$. Тогда достоверность вывода на основе этого правила будет определяться следующим образом. При связи типа И:

$$t_{вывода} = min\{t_{зак}, t_{правила}\} = min\{min\{t_x, t_y\}, t_{правила}\}, \quad (11)$$

а при связи типа ИЛИ:

$$t_{вывода} = min\{t_{зак}, t_{правила}\} = min\{max\{t_x, t_y\}, t_{правила}\}. \quad (12)$$

Базу знаний, учитывающую соотношения нечеткой логики (11) и (12), можно представить в виде И-ИЛИ-графа и описать следующей системой предикатов:

- $node(N, Ls)$ – произвольный узел графа, N – номер узла, Ls – список параметров этого узла;
- $termNode(N)$ – метка для терминальных узлов;
- $goalNode(N)$ – метка для целевых узлов;
- $imp(T, Nz, Ps, C)$ – импликации, соединяющие узлы: T – тип соединения (И, ИЛИ), Nz – узел-заключение, Ps – список узлов-посылок, $C = \overline{0,1}$ – коэффициент субъективной уверенности.

Список узлов-посылок состоит из пар $pos(Np, Sp)$, где Np – посылка, Sp –знак посылки.

1.7. Универсальный решатель нечетких задач

Универсальный решатель нечетких задач УРНЗ подробно описан в работе автора [11]. По соображениям объема приведем лишь основные принципы. Программа УРНЗ написана на языке SWI-Prolog с использованием стандартного предиката *findall(X, infer(N,X), Ls)*, где *infer(N,Ct)* – предикат логического вывода результирующей степени уверенности *Ct* (выходной параметр) для узла *N* (входной параметр). В составе *findall* результат вывода помечен внутренней переменной *X*, что позволяет проследить влияние на Ct всех узлов графа, связанных с *N*. В процессе вывода предикат *infer()* использует соотношения (11), (12).

2. Унифицированный процесс разработки компьютерных обучающих систем

ИОС относятся к классу обучающих систем, называемых по английски как *Intelligent Tutoring Systems* – автоматизированные интеллектуальные обучающие системы, функционирующие без помощи человека (преподавателя). Для эффективного обучения такие системы имеют поддержку мультимедиа, и даже гипермедиа, [21]. Программная сложность таких систем привела к тому, что первенство разработки захватили преимущественно программисты, а дидактическая культура разработчиков остается низкой. Поэтому, несмотря на большое количество уже созданных компьютерных обучающих систем, **существенного влияния на учебный процесс они не оказывают**. Пренебрежение дидактикой приводит к тому, что большинство созданных программ, по сути дела, являются просто демонстрационными, а не инструментами обучения. В этом разделе рассмотрим, как вести программную разработку ИОС на основе нечеткого деятельностного подхода начиная с *дидактического проектирования*.

Основные составляющие процесса обучения, которые необходимо учесть при разработке ИОС: *цель обучения* (зачем учить), *содержание обучения* (чему учить) и *методология обучения* (как учить). Рассмотрим эти составляющие.

При *нечетком* деятельностном подходе изменяются в основном модели, связанные с представлением и использованием *нечетких знаний*, с оперированием ими методами искусственного интеллекта. *Цели*, *содержание* и *методология* обучения остаются при этом неизменными. Поэтому дальнейшее рассмотрение дидактического проектирования ИОС опирается не только на идеи автора, но и, в первую очередь, на работы Г. А. Атанова [3], Б. Ц. Бадмаева [22], П. Я. Гальперина [23], А. Н. Леонтьева [24], Е. И. Машбица [25], Н. Ф. Талызиной [26], Д. Б. Эльконина [27].

2.1. Дидактический проект

Согласно принципам программной инженерии унифицированный процесс разработки сложных программных продуктов (ПП), таких как ИОС, проходит четыре стадии: *анализ и планирование требований*, *проектирование архитектуры*, *программирование*, *пробная пользовательская эксплуатация*, [28]. И все эти четыре стадии повторяются циклически: в каждом цикле снимаются риски, наращивается функциональность. Особенно важна первая стадия цикла – *анализ и планирование требований*, определяющая всю остальную разработку ПП. За детальную разработку программных требований отвечают программисты-аналитики. Но разработка обучающей системы – дело непривычное для

программистов, поскольку для эффективного использования разрабатываемого программного продукта в учебном процессе она должна опирается на слабо знакомую программистам науку дидактику. Поэтому качественно выполнить начальный этап разработки – стадию *анализа и планирования требований*, для профессиональных программистов-аналитиков довольно трудно. Как следствие, они обращают слабое внимание на дидактику и неоправданно высокое – на гипермедиа и другие знакомые программистам технологии. Именно пренебрежением дидактикой объясняются многие неудачи создания обучающих систем для практической педагогики.

Думается, что цикл унифицированного процесса разработки *компьютерных обучающих систем* должен состоять не из четырех, а из пяти стадий: этапу *анализа и планирования требований* должен предшествовать ***этап дидактического проектирования***, выполняемый с участием специалистов по дидактике. Задача этого этапа – построить стратегию и тактику обучения, создать дидактический фундамент для программной инженерии, включающей все остальные этапы разработки программного продукта. Такой дидактический фундамент должен содержать два основных аспекта обучения: *проектирование учебной деятельности* и *управление учебной деятельностью*. Как и все остальные этапы унифицированного процесса, он должен быть итеративным и выполняться в каждом новом цикле совершенствования обучающей системы.

К проектированию учебной деятельности отнесем: 1) *проектирование целей*, 2) *проектирование содержания*, 3) *проектирование технологий обучения*, с помощью которых усваивается содержание, 4) *проектирование системы контроля*, т. е. все, что относится к *моделирование обучаемого*.

Рассмотрим содержание этапа *Дидактическое проектирование* применительно к ИОС с нечеткой адаптивной моделью обучаемого.

2.2. Проектирование модели обучаемого

Используем здесь основные выводы раздела 1.4. Можно рассматривать четыре вида моделей обучаемого: *стартовую*, *нормативную* (эталонную, целевую), *текущую* и *модель ошибок*. Цель обучения задает *нормативная* (эталонная) *модель*, с которой связаны стандарты образования. *Стартовая модель* фиксирует характеристики обучаемого перед процессом обучения, прежде всего, его начальные знания и умения. *Текущая модель* характеризует свойства обучаемого в текущей момент. *Модель ошибок* показывает, как *может* ошибаться обучаемый и какие характеристики обучаемого вытекают из его «ошибок». При разработке обучающей системы вначале необходимо построить и формализовать желаемый образ обучаемого, его целевую, эталонную модель.

Эталонная модель обучаемого является движущей силой учебного процесса: на каждом шаге обучения текущая модель обучаемого сравнивается с эталонной. Эталонная модель должна соответствовать стандартам обучения, разработка которых является одной из главных задач высшей школы. При нечетком деятельностном подходе предметная модель обучаемого выстраивается на основе нечетких знаний и умений обучаемого в рамках изучаемого курса. Компьютерная обучающая система требует формализации представления об обучаемом. В

рассматриваемой ИОС такая формализация опирается на теорию нечетких множеств (см. раздел 1.4.3).

Текущая модель обучаемого и *модель ошибок* являются динамическими и технологически наиболее сложными. *Текущая модель* предназначена для слежения за успехами и неудачами обучения конкретного обучаемого и должна быть *адаптивной*. Механизмами построения текущей модели являются *диагностика* и *тестирование*. Текущая модель является поведенческой моделью обучаемого, и только в простейшем случае ее траекторию должен жестко формировать учитель, считая, что отклонения обучаемого являются ошибочными.

В более совершенных методологиях обучения, развиваемых в последнее время, так называемых *моделях ошибок*, ошибки обучаемого анализируются и даже определяют альтернативные траектории текущей модели, иногда более совершенные, чем те, которые жестко заданы учителем. Недаром говорят: «На ошибках учатся!». Модель ошибок представляет собой дерево движения к цели (к эталонной модели обучаемого), и пути движения по этому дереву определяются ошибками обучаемого. В данной работе мы не рассматриваем модели ошибок, ограничиваясь оверлейной моделью обучаемого.

В начале проектирования, необходимо уяснить, что является целью деятельности обучаемого и обучающей системы, каков должен быть ее конечный результат. Затем необходимо выполнить декомпозицию – провести структурирование деятельности, т. е. определить действия, которые система должна выполнить с обучаемым, благодаря которым будет достигнута цель обучения. Действия можно детализировать с помощью конкретных операций взаимодействия системы с обучаемым.

Поскольку выполнение деятельности и действий обучаемым обусловлено знаниями, то необходимо определить, какими знаниями должна быть наделена система, чтобы обучаемый мог с их помощью выполнять текущую деятельность и действия. При этом надо определить, какие знания обеспечивают реализацию общей ориентировки обучаемого, какие предназначены для исполнительной и какие – для контрольно-корректировачной части.

2.3. Текущая модель обучаемого

Текущее моделирование обучаемого является предметом серьезных исследований искусственного интеллекта в обучении, активно развивающимся в настоящее время в связи с созданием компьютерных обучающих систем. Общий подход к построению текущей модели обучаемого развит в [3, 13, 14]. Текущая модель обучаемого – это элемент процесса обучения в ИОС, определяемый как *учебной деятельностью обучаемых*, так и *деятельностью обучающего*. База знаний и решатель задач в ИОС в схеме рис. 1 построены на основе рассмотренной выше оверлейной модели обучаемого при нечетком деятельностном подходе.

Оверлейные модели хорошо зарекомендовали себя в интеллектуальных обучающих системах при обучении понятиям. Однако они оказались недостаточно эффективными при обучении умениям. Для повышения эффективности диагностики умений в [29] была использована модель диагностики на основе *генетического графа*. Его узлы представляют собой элементы декларативных и процедурных знаний, описывающих возможные *умения* обучаемого, а дуги –

отношения обобщения, конкретизации и отклонения. В отличие от типичной оверлейной модели, генетический граф как текущая модель, содержит не только умения, предусмотренные нормативной моделью, но и различные обобщения, конкретизации и отклонения от эталона, которые определяются действиями обучающего. Текущей модели конкретного обучаемого соответствует его путь (траектория) на генетическом графе.

2.3.1. Проектирование целей

Человек учится для того, чтобы потом работать по избранной специальности. Конечной целью обучения является не вооружение знаниями, не накопление их, а формирование умения работать, правильно действовать при решении проблем, освоение способов действий. Знаниям можно научиться только в процессе их использования в деятельности, только оперируя ими, [24]. Это – основные принципы формулирования цели при деятельностном подходе. При *нечетком* деятельностном подходе специфичной будет только их технологическая реализация.

2.3.2. Проектирование содержания

Конкретная задача обучения – усвоение *содержания обучения*. Если при традиционном обучении содержанием обучения была система знаний (теория, научная информация), то при деятельностном обучении содержанием обучения становится *заданная система действий* обучаемого. При этом знания становятся не целью обучения, а средством. Знания усваиваются лишь для того, чтобы с их помощью выполнять действия. Знаниям отводится не основная, а поддерживающая роль: с их помощью нужно сформировать у обучаемых *умение осуществлять деятельность*, выстроить способы действий. Таким образом, *содержание обучения составляют: 1) заданная система действий и 2) те знания, которые обеспечивают освоение этой системы действий*, [3].

Заданная система действий – это полный набор действий, с помощью которого достигается заявленная цель изучения курса при деятельностном подходе. Каждое действие состоит из конкретных операций, и чтобы их выполнить, нужны умения. Обучение действиям означает формирование умения решать задачи (проблемы). Формирование умения не сводится, однако, к конкретным методическим рецептам. Методика оперирует материалами конкретных дисциплин, это – уровень рецептов по конкретным умениям, тогда как дидактика – общая для всех дисциплин методология, системно обучающая рациональным способам действий при решении *любых* проблем. Логическое построение системы действий должно быть воплощено аналитиком-программистом в конкретные функциональные требования к ИОС.

2.3.3. Проектирование технологий обучения

Методология обучения при деятельностном подходе основана на представлении об обучаемом. Ее можно охарактеризовать как личностную и адаптивную. Вначале, исходя из стандартов обучения, создается эталонная *модель обучаемого*. Затем путем предварительного тестирования формируется стартовая модель *конкретного* обучаемого, и весь процесс обучения представляет собой движение от стартовой модели к эталонной как пошаговое научение индивида. Каждый шаг сопровождается тестированием и адаптивным приближением

текущей модели обучаемого к эталонной. Когда текущая модель достигает эталонной, цель обучения считается выполненной и процесс останавливается.

Наиболее сильно Нечеткий деятельностный подход отразится на методологии обучения. При сохранении общих принципов описанной выше методологии обучения, принципы обработки нечетких знаний методами искусственного интеллекта вступят здесь в полную силу. Это связано с тем, что искусственный интеллект обладает эффективными методами обработки нечетких знаний и постоянно их совершенствует. Выше, при рассмотрении процесса обучения с нечеткой оверлейной моделью обучаемого, мы продемонстрировали использование в обучении некоторых методов искусственного интеллекта, в частности, основанных на нечетких множествах.

2.3.4. Проектирование системы контроля

Система контроля предназначена для определения конечных результатов обучения, поэтому она должна быть не диагностической, а тестирующей. В случае деятельностного подхода в обучении можно утверждать, что если студент умеет что-то делать, то он знает и как это делать, т. е. владеет знаниями. Поэтому достаточно проверить только умения.

Проектируя систему контроля в ИОС на основе деятельностного подхода, следует придерживаться определенных принципов, несколько отличающихся от принципов, сформулированных в педагогике, [3, 30]. Прежде всего, необходимо ответить на вопрос: «Что означает знать предмет при деятельностном подходе в обучении?». Ответ находится в предметной нормативной модели обучаемого. К сожалению, общепринятые тесты с трудом справляются с задачами проверки обучаемого, на предмет того, как он овладел учебной деятельностью.

2.4. Управление учебной деятельностью

Механизмом обучения является не передача готовых знаний обучаемому учителем, а *управление учебной деятельностью*. Обучающий направляет и корректирует учебную деятельность обучаемого, *управляет* ею для достижения учебных целей. Существуют различные способы управления учебной деятельностью. Наиболее полно обобщает и классифицирует учебную деятельность искусственный интеллект в обучении, предлагая самые совершенные методы управления. Нечеткий деятельностный подход целиком и полностью опирается на методы искусственного интеллекта в обучении.

Классификация методов управления учебной деятельностью приведена в работах [13, 14]. Разрабатывая стратегию процесса обучения в ИОС необходимо подробно рассмотреть эту и другие подобные классификации, т. к. управление учебной деятельностью порождает основные функциональные требования для следующего этапа Унифицированного процесса – этапа Анализа и планирования требований. В данной работе нет возможности описать методику получения требований, хотя в стадии дидактического проектирования это должно быть сделано обязательно.

Как мы уже упоминали, одной из важных психологических задач обучения является реакция на неизбежное забывание обучаемым ранее достигнутых умений. В [3] обсуждается модель

процесса забывания умений и ее приложение к оверлейной модели обучаемого. Управление учебной деятельностью основано на использовании *агенда*-механизма. *Агенда* (от англ. agenda) – это упорядоченный список задач, последовательность операций. В процессе работы система формирует *агенды* из всех задач, выданных обучаемому при тестировании, решение которых требует использования заданных умений. Затем производится *выбор* задачи, в наибольшей степени подходящей для конкретного обучаемого.

Деятельность – это решение задач. Для решения задачи обучаемый должен выполнить ряд последовательных *действий*, каждое из которых обусловлено своим мотивом, своей сознательно поставленной подцелью. Действия, в свою очередь, состоят из *операций*, представляющих способы осуществления действий, реализующих подходящие технологии обработки нечетких знаний методами искусственного интеллекта.

Таким образом, модель учебной деятельности включает *задачи*, *действия* и *операции*. Решая задачу, обучаемый выполняет некоторые действия. *Действия* – это способ решения задачи, *механизм* осуществления деятельности. Действия, в свою очередь, состоят из конкретных *операций*. В целом выстраивается следующая цепочка, характеризующая деятельность: *мотив* → *цель* → *подцели* → *задачи* → *действия* → *операции* → *продукт*, [24]. По мере осуществления деятельности обучаемый решает задачи, одну за другой. Двигаясь по изучаемому курсу в соответствии с поставленной целью, обучаемый выполняет переходы от одной подцели к другой, и каждая задача связана с очередной подцелью обучения. Обучаемый выполняет операции опираясь на *средства* осуществления *учебной* деятельности. *Учебная деятельность* обучаемого в высшей школе состоит в том, чтобы освоить способы действий, на которых будет основываться его будущая профессиональная работа, а ИОС является эффективным инструментом учебной деятельности обучаемого. При этом для учебной деятельности обучаемого важен не результат решения задачи (ответ), а *процесс* получения результата.

Поскольку ИОС является средством обучения, то проектирование учебной деятельности касается ее непосредственно. Детали мы не будем обсуждать в настоящей работе, отметим только, что при деятельностном подходе к обучению *структурирование учебной деятельности* может быть проведено с нескольких точек зрения: *функциональной*, *динамической*, *операционной* и *организационной*.

Единицей деятельности является *действие*: учебная деятельность обучаемого состоит из последовательно выполняемых им действий по решению поставленных учебных задач. В составе действия можно выделить основные функциональные части: *мотивационную, ориентировочную, содержательную, исполнительную* и *контрольно-корректировочную*. Согласно исследованиям [24], умственные действия развиваются поэтапно:

- первый этап – *мотивационный*: действие еще не выполняется, оно только подготавливается;
- второй – *выполнение* действия с развертыванием всех входящих в него операций;
- третий – *речевой*: обучаемый проговаривает перед обучающим все операции действия, а обучающий, если надо, поправляет его;
- четвертый – *мысленный*: обучаемый обдумывает мысленно и проговаривает (про себя) все операции, как и в предыдущем этапе;

- пятый – *превращение в навык*: действие сжимается и автоматизируется, превращаясь в навык.

Усвоение учебного материала на высшем уровне (на отлично) предполагает *понимание* знаний, их *запоминание* и возможность *активного использования*. Деятельность самого обучающего включает при этом три основных аспекта, которые положены в основу разработки ИОС:

- *проектирование* учебной деятельности обучаемого;
- *организация* и *обеспечение* его учебной деятельности;
- *управление* учебной деятельностью обучаемого.

Проектирование учебной деятельности включает проектирование целей, содержания, технологий обучения, с помощью которых усваивается содержание, системы контроля. После проектирования целей обучения, необходимо определить состав деятельности обучаемого, т. е. действия и составляющие их операции, с помощью которых цель достигается. ИОС выступает при этом в качестве средства, с помощью которого организуется учебная деятельность. Проектирование содержания и контроля учебной деятельности относится к моделированию обучаемого при прохождении учебного процесса.

Несмотря на описание множества идей управления процессом обучения, до сих пор не сложилось общепринятых рекомендаций по управлению учебной деятельностью в ИОС. Лучшие подходы основаны на моделировании поведения человека, используемые в искусственном интеллекте в обучении. Но в компьютерных обучающих системах такие модели должны быть поняты и описаны чрезвычайно детально, формализованы до такой степени, чтобы лечь в основу получения детальных требований этапа «Анализ и планирование требований» Унифицированного процесса, и быть потом запрограммированными. Так что разработка стадии Дидактического проектирования Унифицированного процесса, выполняемая содружеством дидактиков, программистов и специалистов по искусственному интеллекту в основном еще впереди.

Подведем итоги. Для разработки *программных проектов обучающих систем* предлагается *пятистадийный* ***«Унифицированный процесс разработки компьютерных обучающих систем»***, в котором дополнительная стадия *«Дидактическое проектирование»* предшествует стадии «Анализа и планирования требований» типового в программной инженерии *четырехстадийного* Унифицированного процесса разработки программных продуктов. Стадии Унифицированного процесса разработки компьютерных обучающих систем:

1. Дидактическое проектирование.
2. Анализ и планирование требований.
3. Проектирование архитектуры.
4. Реализация (собственно программирование).
5. Внедрение (пробная пользовательская эксплуатация).

Основные положения и содержание стадии *Дидактическое проектирование*:

- механизмом осуществления учебной деятельности является *решение задач* (проблем);
- конечной целью обучения является *формирование способа действий*: учитель или обучающая система должны научить обучаемого умению выполнять действия и операции, с помощью которых решается задача;

- знаниям в процессе обучения отводится не центральная роль: они являются *средством*, которое нужно только для правильного выполнения действий по решению задачи;
- *учебная деятельность* обучаемого планируется и организуется обучающим;
- *цель* учебной деятельности обучаемого задается обучающим (возможно и самообучение): обучаемому даются задачи и его цель – решение этих задач;
- *продукт* учебной деятельности – изменение самого обучаемого (был неучем, стал профессионалом);
- *прямой продукт* учебной деятельности – изменение обучаемого, в соответствии с целью курса;
- *побочный продукт* учебной деятельности – изменение обучаемого, не предусмотренные целью курса (встречается при самообразовании, а также в методологиях обучения типа «модель ошибок» и т. п.);
- *ядром и существом учебной деятельности обучаемого является решение учебных задач*, т. е. задач, направленных на достижение учебных целей;
- главным при решении учебной задачи является *не ответ*, *а процесс его получения*, способ решения задачи.

Широкое использование нечеткой логики и нечетких множеств позволяет ввести мощный унифицированный формат описания знаний на основе логики предикатов ППП, дополненный описанием нечетких знаний. Такое однородное описание упрощает создание единой базы знаний, однотипно описывающей все знания: дидактические, адаптивные, управляющие, а также дает возможность построить *универсальный* решатель нечетких задач обучения на языке Пролог.

Мы рассмотрели этап *дидактического проектирования* ИОС с довольно общих позиций. В процессе реальной разработки этот важнейший этап создания ИОС должен заканчиваться выпуском документа, известного в зарубежной практике IT-разработчиков как *Vision (Видение)*. Этот выходной артефакт этапа дидактического проектирования будет отражать *документальное* видение ИОС с позиции дидактика, что очень важно для последующих этапов разработки программного проекта.

2.5. Двухслойная многопотоковая архитектура ИОС

Для программной реализации ИОС была разработана *двухслойная многопотоковая* архитектура. Наружный слой системы (главный поток) служит для организации управления и прокладки оптимального пути по «дебрям» курса в соответствии с целями и индивидуальными особенностями обучаемого. Внутренний слой (вторичные потоки, обрабатывающие информацию в фоновом режиме) отвечает за адаптацию модели обучаемого. Такая двухслойная организация позволяет разделить функциональную ответственность программных модулей, повысить устойчивость архитектуры к изменению дидактической модели процесса обучения и увеличить производительность системы. С помощью модели процесса обучения, построенной на нечеткой логике и нечетких множествах, удается хорошо согласовать программную архитектуру наружного и внутреннего слоев системы. Благодаря общей базе знаний и универсальному решателю задач, система открыта для продолжающихся исследований без серьезного перепрограммирования.

2.6. Диалог в ИОС: программная архитектура обратных вызовов клиента

Как отметил П. Л. Брусиловский [1], развитие сетевых ИОС сильно тормозят проблемы диалога сервера, решающего задачу обучения, с вызвавшим его клиентом (браузером). Особенно это касается гибридных ИОС, в которых основная логика процесса обучения выполняется модулем, написанном на Прологе и размещенном на сервере. Поскольку сам принцип логического программирования на Прологе требует постоянного хранения в оперативной памяти альтернативных вариантов решения задачи, связанных с бэктрекингом, то мы должны запрашивать с сервера у клиента нужные данные для процесса решения во время одной сессии, *не прерывая хода выполнения пролог-программы*. Суть проблемы в том, что браузер (клиент) может свободно посылать запросы удаленному серверу, а вот у сервера возникают большие проблемы с обращением к браузеру, т. к. по HTTP-протоколу с ответом сервера связь прекращается до новой очередной сессии. Это затрудняет накопление на серверной стороне информации о текущем состоянии обучаемого, оптимизацию последовательности и адаптации во время сессии.

Проблема обратных вызовов клиента со стороны сервера до недавнего времени была почти непреодолимым препятствием на пути решения любых неформализованных задач методами искусственного интеллекта в клиент/серверных системах с удаленным сервером. В настоящее время, когда новый интернет-протокол HTTP/1.1 уже шагает по планете, когда появились такие новинки интернет-технологий как AJAX, сокеты и др., решение проблемы обратных вызовов становится возможным. Опишем здесь один из способов решения этой проблемы с использованием PHP – популярного языка web-программирования.

Предложенное решение использует как новейшие возможности в области взаимодействия со стороны PHP, так и последние разработки в этом направлении со стороны Prolog. Такая связка двух технологий организуется на основе библиотеки cURL со стороны PHP и библиотек поддержки передачи данных по протоколу HTTP со стороны SWI-Prolog. Предложенное решение обеспечивает возможность запуска Prolog-программы с помощью HTTP-запроса, а также взаимодействия с ней клиентского приложения. Относительно PHP архитектура обратных вызовов представляет собой клиента, основанного на библиотеке cURL, а в отношении Prolog – многопоточный http-сервер, который позволяет запускать несколько Prolog-программ в разных потоках выполнения и взаимодействовать с ними нескольким клиентам одновременно. Конечно, решение пока что является ресурсозатратным и требует оптимизации, но уже сейчас оно позволяет обеспечить определенную масштабируемость вычислений. Основным форматом сообщений, передаваемых между клиентским приложением на PHP и серверной частью на Prolog, является формат JSON: он позволяет снизить издержки на кодирование сообщения, а также является легко понятным форматом не только для вычислительной системы, но и для человека, что значительно упрощает отладку и тестирование обратных вызовов. Относительно PHP система обратных вызовов представляет собой набор классов, базирующихся на библиотеке cURL и функциях обработки формата JSON. Такая архитектура позволяет формировать и отправлять запросы-сообщения к серверному приложению на Prolog. Сообщения могут быть нескольких типов:

- вызов определенной Prolog-программы;
- передача значения переменной;

- вызов определенного предиката с несколькими аргументами;
- завершение работы программы.

Сообщения данных типов интерпретируются и выполняются на стороне Prolog, что позволяет достаточно гибко взаимодействовать с Prolog-программой. Интерпретация сообщения на стороне Prolog'а проходит следующие этапы:

1. Получение HTTP-запроса.
2. Извлечение данных из формата JSON и определение типа сообщения.
3. Задание направления передачи данных: передача потоку выполнения данных сообщения, запуск Prolog-программы либо ее завершение.
4. Определение потока выполнения (если он уже был инициализирован), его инициализация либо завершение.
5. Получение результатов, либо нового запроса данных для клиента.
6. Формирование сообщения для клиента.
7. Отправка сообщения клиенту как ответ на HTTP-запрос.

Рассмотрим эти этапы подробнее.

Получение HTTP-запроса. На данном этапе производится получение запроса, выделение сообщения и аутентификация клиента. Этап базируется на последних нововведениях в SWI-Prolog, таких как возможность создания HTTP-сервера.

Извлечение данных из JSON. На этом этапе производится преобразование данных в формате JSON в стандартные предикаты Prolog, что позволяет выделить из них данные и определить тип сообщения. Тип сообщения необходим для выбора действия, которое будет выполняться конкретной Prolog-программой или сервером.

Задание направления передачи данных. Задание направления передачи данных: передача потоку выполнения данных сообщения, либо запуск Prolog-программы. Как уже говорилось ранее, требуется определение типа сообщения, чтобы локализовать действия, сведения о которых передаются в сообщении. Это связано с архитектурой предлагаемого решения на Prolog. Некоторые действия, необходимые для корректной работы данного решения, не могут быть выполнены непосредственно в Prolog-программе, так как носят скорее системный характер (например, запуск и остановка потоков выполнения), поэтому требуют использования потока выполнения сервера. Именно с этим связана необходимость определения типа и последующего направления передачи данных. То есть на данном этапе организуется ветвление, которое определяет, как серверу действовать дальше: или выполнить действия самому, или передать данные, полученные в сообщении, дальше определенному потоку выполнения.

Определение потока выполнения. Данный этап необходим для определения идентификатора потока выполнения в Prolog, либо инициализации нового потока для выполнения Prolog-программы. Возможны различные варианты реализации данного этапа. Первый вариант – это передача идентификатора через клиента: при инициализации Prolog-программы в новом потоке исполнения клиент получает его идентификатор и все последующие вызовы производятся с включением в сообщение этого идентификатора. Второй вариант – это хранить

данные о потоках на стороне Prolog, идентифицируя их какой-либо информацией о клиенте. Для реализации этого метода Prolog предоставляет несколько возможных решений, одно из них – динамические предикаты. Предпочтительным является второй вариант, так как позволяет уменьшить связность между клиентом на PHP и сервером на Prolog, что значительно повышает отказоустойчивость всей системы. Однако первый способ несколько легче в реализации. Инициализация потока базируется на стандартных средствах Prolog в области обеспечения многопоточности. Но и здесь есть некоторые детали, о которых стоит упомянуть. В первую очередь это то, что потоки в промежутках времени между сообщениями должны находиться в режиме ожидания, это неизбежно накладывает некоторые ограничения на исполняемую в потоке программу, так как реализовать такое поведение не всегда представляется возможным, см. ниже.

Получение результатов, либо нового запроса данных для клиента. Как во время, так и после окончательного выполнения программы существует необходимость не только запрашивать данные у клиента, но и отправлять промежуточные сведения о ходе выполнения программы на стороне Prolog, чтобы выполнить их визуализацию на стороне клиента. Используя для этого предлагаемую архитектуру обратных вызовов, можно встраивать в сообщения от Prolog-программ не только запросы на получение данных от клиента, но и любые произвольные данные, необходимые для обработки на клиентской стороне. Запросы вместе с передаваемыми данными формируются на стороне Prolog-программы в виде предикатов и затем передаются серверу. Он оформляет их в сообщение в формате JSON и передает клиенту, который затем может их использовать в целях обработки и предоставления пользователю или в иных целях. Данный подход действует как для запросов на получение данных от клиента, так и для результатов работы программы (промежуточных и конечных).

Формирование сообщения для клиента. Данный этап представляет собой процесс преобразования Prolog-предикатов в сообщения, которые можно передавать через глобальную сеть. Сообщения содержат в себе данные и запросы, полученные от потока выполнения Prolog-программы и некоторую служебную информацию, необходимую для идентификации сервера и типов данных, полученных от него. Все это сообщение преобразуется в формат JSON и отправляется клиенту в виде ответа на HTTP-запрос.

Отправка сообщения клиенту. Этот этап представляет собой чисто технический процесс передачи данных между клиентом и сервером, поэтому не имеет смысла описывать данный процесс в этой статье.

2.6.1. Процесс на стороне клиента

На стороне клиента процесс выглядит достаточно просто. После получения данных клиентское приложение преобразует их в наиболее удобный для себя вид и использует полученные данные так, как будет необходимо. Хочется отметить, что взаимодействие клиента и сервера не является потоковым и прерывается, как правило, со стороны клиента, но подобное взаимодействие является достаточно распространенным в глобальной сети, очень отработанным, надежным и универсальным. Также можно отметить тот факт, что в качестве клиента может выступать не только приложение, написанное на PHP, но и чисто клиентские сценарии, выполняемые непосредственно в браузере пользователя, однако для этого

представленная архитектура требует доработки до полноценного HTTP-сервера. Что касается масштабируемости данной архитектуры, она может быть реализована как за счет количества выполняемых одновременно потоков и поддерживаемых соединений, так и наращивания аппаратной мощности физической платформы, на которой данному решению предстоит работать. Простота организации клиентской части приложения позволяет обойтись «тонким» клиентом, работающим с любым из популярных браузеров. Описанная архитектура доведена до уровня законченного web- приложения [11].

Предложенная программная архитектура выполнения обратных вызовов клиента со стороны сервера эффективно решает проблему быстродействующего и масштабируемого диалога в клиент/серверных системах для организации решения неформализованных задач методами искусственного интеллекта в диалоге с ЭВМ, в том числе и задач обучения.

Заключение

Основные результаты работы:

1. Предложен ***Нечеткий деятельностный подход*** к обучению как развиваемый в дидактике деятельностный подход, в котором модель обучаемого и процесс обучения основаны на нечетких знаниях и методах искусственного интеллекта. При таком подходе использование нечеткой логики и нечетких множеств позволяет ввести мощный унифицированный формат описания знаний на основе логики предикатов ППП, дополненный описанием нечетких знаний. А это приводит к более совершенным моделям обучаемого и процессам обучения, улучшенным механизмам адаптации и более эффективным компьютерным обучающим системам, особенно web-ориентированным и масштабируемым.
2. Проведено широкое использование *нечеткой логики* и *нечетких множеств в моделях обучающей системы,* в частности, представление оверлейной модели обучаемого в виде нечеткого множества пар. Такая «стандартизация» описания дидактических моделей облегчает формирование базы знаний и решателя задач разработчикам-программистам.
3. Введена *матрица релевантности концептов* для связи эталонной модели курса с текущей предметной моделью обучаемого и последующей адаптацией модели обучаемого по результатам тестовых заданий проверки навыков/умений. Матрица релевантности упрощает нахождение нечеткого соответствия действий обучаемого нормативным концептам учебного материала.
4. Разработана программная архитектура *обратных вызовов клиента* со стороны удаленного сервера для решения неформализованных задач обучения методами искусственного интеллекта в диалоге с ЭВМ.
5. Для повышения педагогического качества разработки ИОС введен *пятистадийный* ***Унифицированный процесс разработки компьютерных обучающих систем***, в котором дополнительная стадия *«Дидактическое проектирование»* предшествует стадии «Анализ и планирование требований» типового в программной инженерии четырехстадийного Унифицированного процесса разработки программных проектов. Принципы и содержание введенной стадии дидактического проектирования

рассмотрены на примере программного проекта ИОС с оверлейной моделью обучаемого и Нечетким деятельностным подходом в обучении.

Полная теория сетевой ИОС на основе Нечеткого деятельностного подхода с адаптивной моделью обучаемого еще далеко не завершена. Многое только обозначено и требует развития, но основные контуры уже прорисовываются. Ситуация, сложившаяся в области компьютерного обучения, является парадоксальной: несмотря на активно ведущиеся поиски, обилие результатов, ощущается необходимость кардинальных изменений концепции обучения, подхода к компьютерному обучению. В первую очередь, требуется разработка теории компьютерного обучения, новых методов представлений знаний, моделирования процесса деятельностного обучения и поведения обучаемого, развития Унифицированного процесса разработки компьютерных обучающих систем. С внедрением компьютерного обучения стали меняться стили и устоявшиеся подходы к обучению. Трудно переоценить значение и влияние этих изменений на сферу образовательной деятельности человека в целом.

Список литературы

1. Брусиловский П.Л. Адаптивные интеллектуальные технологии в сетевом обучении // Новости искусственного интеллекта. 2002. № 5. С. 25-31.
2. Рыбина Г.В. Обучающие интегрированные экспертные системы: некоторые итоги и перспективы // Искусственный интеллект и принятие решений. 2008. № 1. С. 22-46.
3. Атанов Г.А., Пустынникова И.Н. Обучение и искусственный интеллект, или основы современной дидактики высшей школы / Под ред. Г.А. Атанова. Донецк: Издательство ДОУ, 2002. 504 с.
4. Галеев И.Х. Модель управления процессом обучения в ИОС // Образовательные технологии и общество. 2010. Т.13. № 3. С. 285-292.
5. Удальцов С.В. Среда разработки и поддержки систем дистанционного обучения IDLE: технологический аспект // Дистанционное образование. 1998. № 1. С. 44-48.
6. Решетова З.А. Процесс усвоения как деятельность // Современные проблемы дидактики высшей школы: сб. избр. трудов междунар. конф. Донецк: Изд-во ДонГУ, 1997. С. 3-12.
7. Лорьер Ж.-Л. Системы искусственного интеллекта : пер. с франц. М.: Мир, 1991. 356 с.
8. Попов Э.В. Экспертные системы: Решение неформализованных задач в диалоге с ЭВМ. М.: Наука, 1987. 288 с.
9. Астанин С.В. Сопровождение процесса обучения на основе нечеткого моделирования // Дистанционное образование. 2000. № 5. Режим доступа: http://www.e-joe.ru/sod/00/5_00/as.html (дата обращения 19.11.2012).
10. Клюкин В.Э. и др. Принятие решений в советующих партнерских системах при нечеткой исходной информации // Труды Всеросийской конференции «Новые информационные технологии в исследовании дискретных структур». Екатеринбург: УрО РАН, 1996. С. 27-34.

11. Клюкин В.Э. Программирование интеллектуальных систем на Microsoft Visual C++ .NET. Кн. 2. Интеграция на основе COM, поддержка в ATL и MFC. Екатеринбург: УГТУ-УПИ, 2007. 334 с.
12. Клюкин В.Э. Композиция ГЛОТЕОН: глоссарий + тезаурус + онтология // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2011. № 9. Режим доступа: http://technomag.edu.ru/doc/206071.html (дата обращения 19.11.2012).
13. Dillenbourg E., Self J. Framework for Lerner Modeling // Interactive Learning Environments. 1992. Vol. 2, Iss. 2. P. 111-137.
14. Murray W.R. Control for Intelligent Tutoring Systems: A Blackboard-based Dynamic Instructional Planner // Proceedings of the 4th International Conference on Artificial Intellegence and Education. Amsterdam, Nitherlands. 1989. P. 150-168.
15. Niu X. Purpose Based Learner Modelling // Proc. of the Grad Symposium. Canada, Department of Computer Science, University of Saskatchewan, 2002.
16. Коляда М.Г. Виды моделей, обучаемых в автоматизированных обучающих системах // Искусственный интеллект. 2008. № 2. Режим доступа: http://www.nbuv.gov.ua/portal/natural/ii/2008_2/JournalAI_2008_2/Razdel2/00_Kolyada.pdf
17. Петрушин В.А. Экспертно-обучающие системы. Киев: Наукова думка, 1992. 196 с.
18. Растригин Л.А., Эренштейн М.Х. Адаптивное обучение с моделью обучаемого. Рига: Зинатне, 1988. 160 с.
19. Буль Е.Е. Обзор моделей студента для компьютерных систем обучения // Образовательные технологии и общество (Educational Technology & Society). 2003. Том. 6, № 4. С. 245-250. Режим доступа: http://ifets.ieee.org/russian/depository/v6_i4/html/G.html (дата обращения 19.11.2012).
20. Щеголькова В.А., Любчак В.А., Рудень Р.Н. Модель ученика в компьютерных обучающих системах // Вестник Сумского государственного университета. Технические науки. 2003. № 11 (57). С. 35-42.
21. Касьянов В.Н., Касьянова Е.В. Методы и средства адаптивной гипермедиа // Вычислительные технологии. 2004. Т. 9, Ч. 2. С. 333-341.
22. Бадмаев Б.Ц. Психология и методика ускоренного обучения. М.: Владос, 1998. 272 с.
23. Гальперин П.Я. Психология мышления и учение о поэтапном формировании умственных действий // Исследования мышления в советской психологии: сб. науч. трудов. М.: Наука, 1966. С. 236-278.
24. Леонтьев А.Н. Избранные психологические произведения: В 2 т. М.: Педагогика, 1983. Т. 1. 391 с.; т. 2. 318 с.
25. Машбиц Е.И. Психолого-педагогические проблемы компьютеризации обучения. М.: Педагогика, 1988. 192 с.
26. Талызина Н.Ф. Управление процессом усвоения знаний. М.: Изд-во МГУ, 1984. 288 с.
27. Эльконин Д.Б. Избранные психологические труды. М.: Педагогика, 1989. 560 с.

28. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения : пер. с англ. СПб.: Питер, 2002. 496 с. (The Unified Software Development Process)
29. Goldstein I.P. The Genetic Graph: a representation for the evolution of procedural knowledge // International Journal of Man-Machine Studies. 1979. № 11. P. 51-77.
30. Карпова И.П. Исследование и разработка подсистемы контроля знаний в распределенных автоматизированных обучающих системах: дис…. канд. техн. наук. Москва, 2002. 200 с.

SCIENTIFIC PERIODICAL OF THE BAUMAN MSTU
SCIENCE and EDUCATION
EL № FS77 - 48211. №0421200025. ISSN 1994-0408
electronic scientific and technical journal

Web-oriented intelligent training systems based on the fuzzy pragmatist approach in teaching

11, November 2012
DOI: 10.7463/1112.0489620

Klyukin V.E.

Russia, Ekaterinburg, Ural Federal University. Physics and Technology Institute
vt@dpt.ustu.ru
kve2310@gmail.com

Publications with keywords: web-oriented intelligent tutoring systems, fuzzy activity approach in teaching, fuzzy matching relevance of concepts, the relevance matrix, two-layer ITS management organization, multi-threaded software architecture ITS,client callbacks, language embedding SWI-Prolog in PHP, Unified Process development of computer teaching systems
Publications with words: web-oriented intelligent tutoring systems, fuzzy activity approach in teaching, fuzzy matching relevance of concepts, the relevance matrix, two-layer ITS management organization, multi-threaded software architecture ITS,client callbacks, language embedding SWI-Prolog in PHP, Unified Process development of computer teaching systems

The author considers a web-oriented computer training system with a fuzzy adaptive model of a student. This system is based on a modified pragmatist approach in teaching, which was called the fuzzy pragmatist approach by the author; in this approach the model of a student and learning process are based on fuzzy knowledge and methods of artificial intelligence. The author proposed two-layer multi-threaded control for a training system, it consists of the following layers: a layer of top-level (main stream), which is responsible for managing training activities and navigating the course, and internal level (secondary flow, here information is processed in the background) which is responsible for adaptation of the model of a student. A unified development process of computer-based training systems with the stage of a didactic design was introduced. The problem of callbacks from the server to the client was tackled in order to deal with non algorithmic problems occurring during the dialogue with PC by means of methods of artificial intelligence in client/server systems with remote server, including learning tasks.

References

1. Brusilovskii P.L. Adaptivnye intellektual'nye tekhnologii v setevom obuchenii [Adaptive Intellectual technologies in network education]. *Novosti iskusstvennogo intellekta* [News of artificial intelligence], 2002, no. 5, pp. 25-31.

2. Rybina G.V. Obuchaiushchie integrirovannye ekspertnye sistemy: nekotorye itogi i perspektivy [Educational integrated expert systems: some results and prospects]. *Iskusstvennyi intellekt i priniatie reshenii* [Artificial intelligence and decision-making], 2008, no. 1, pp. 22-46.

3. Atanov G.A., Pustynnikova I.N. *Obuchenie i iskusstvennyi intellekt, ili osnovy sovremennoi didaktiki vysshei shkoly* [Learning and artificial intelligence, or the foundations of modern didactics of higher school]. Donetsk, DOU Publ., 2002. 504 p.

4. Galeev I.Kh. Model' upravleniia protsessom obucheniia v IOS [Model of managing the learning process in information - educational environment]. *Obrazovatel'nye tekhnologii i obshchestvo* [Educational technology and society], 2010, vol. 13, no. 3, pp. 285-292.

5. Udal'tsov S.V. Sreda razrabotki i podderzhki sistem distantsionnogo obucheniia IDLE: tekhnologicheskii aspekt [Development environment and support of distance education IDLE: the technological dimension]. *Distantsionnoe obrazovanie* [Distance education], 1998, no. 1, pp. 44-48.

6. Reshetova Z.A. Protsess usvoeniia kak deiatel'nost' [The process of learning as the activities]. *Sovremennye problemy didaktiki vysshei shkoly: sb. izbr. trudov mezhdunar. konf.* [Modern problems of didactics of higher education: a collection of selected works of the international conference]. Donetsk, DonGU Publ., 1997, pp. 3-12.

7. Lor'er Zh.-L. *Sistemy iskusstvennogo intellekta* [Artificial intelligence systems]. Transl. from French. Moscow, Mir, 1991. 356 p.

8. Popov E.V. *Ekspertnye sistemy: Reshenie neformalizovannykh zadach v dialoge s EVM* [Expert systems: the Solution of non-formalized problems in a dialogue with a computer]. Moscow, Nauka, 1987. 288 p.

9. Astanin S.V. Soprovozhdenie protsessa obucheniia na osnove nechetkogo modelirovaniia [Support of the learning process on the basis of fuzzy modeling]. *Distantsionnoe obrazovanie* [Distance education], 2000, no. 5. Available at: http://www.e-joe.ru/sod/00/5_00/as.html , accessed 19.11.2012.

10. Kliukin V.E., et al. Priniatie reshenii v sovetuiushchikh partnerskikh sistemakh pri nechetkoi iskhodnoi informatsii [Decision-making in the advice partner systems with fuzzy initial information]. *Trudy Vserosiiskoi konferentsii «Novye informatsionnye tekhnologii v issledovanii diskretnykh struktur»* [Proceedings of all-Russian conference «New information technologies in the study of discrete structures».]. Ekaterinburg, Ural branch of RAS Publ., 1996, pp. 27-34.

11. Kliukin V.E. *Programmirovanie intellektual'nykh sistem na Microsoft Visual C++ .NET. Kn. 2. Integratsiia na osnove COM, podderzhka v ATL i MFC* [Programming of intelligent systems in Microsoft Visual C++ .NET. Book 2. Integration on the basis of the COM, support in ATL and MFC]. Ekaterinburg, UGTU-UPI Publ., 2007. 334 p.

12. Kliukin V.E. Kompozitsiia GLOTEON: glossarii + tezaurus + ontologiia [Composition GLOTEON: glossary + thesaurus + ontology]. *Nauka i obrazovanie MGTU im. N.E. Baumana* [Science and Education of the Bauman MSTU], 2011, no. 9. Available at: http://technomag.edu.ru/doc/206071.html , accessed 19.11.2012.

13. Dillenbourg E., Self J. Framework for Lerner Modeling. *Interactive Learning Environments*, 1992, vol. 2, no. 2, pp. 111-137.

14. Murray W.R. Control for Intelligent Tutoring Systems: A Blackboard-based Dynamic Instructional Planner. *Proc. of the 4th International Conference on Artificial Intellegence and Education*. Amsterdam, Nitherlands, 1989, pp. 150-168.

15. Niu X. Purpose Based Learner Modelling. *Proc. of the Grad Symposium*. Canada, Department of Computer Science, University of Saskatchewan, 2002.

16. Koliada M.G. Vidy modelei, obuchaemykh v avtomatizirovannykh obuchaiushchikh sistemakh [Kinds of student models in the automated learning systems]. *Iskusstvennyi intellect* [Artificial intelligence]. 2008. № 2. Available at: http://www.nbuv.gov.ua/portal/natural/ii/2008_2/JournalAI_2008_2/Razdel2/00_Kolyada.pdf , accessed 19.11.2012.

17. Petrushin V.A. *Ekspertno-obuchaiushchie sistemy* [Expert-learning systems]. Kiev, Naukova dumka, 1992. 196 p.

18. Rastrigin L.A., Erenshtein M.Kh. *Adaptivnoe obuchenie s model'iu obuchaemogo* [Adaptive learning with the student model]. Riga, Zinatne, 1988. 160 p.

19. Bul' E.E. Obzor modelei studenta dlia komp'iuternykh sistem obucheniia [Overview of the student models for computer-assisted learning systems]. *Obrazovatel'nye tekhnologii i obshchestvo* [Educational Technology & Society], 2003, vol. 6, no. 4, pp. 245-250. Available at: http://ifets.ieee.org/russian/depository/v6_i4/html/G.html , accessed 19.11.2012.

20. Shchegol'kova V.A., Liubchak V.A., Ruden' R.N. Model' uchenika v komp'iuternykh obuchaiushchikh sistemakh [Student model in computer-assisted learning system]. *Vestnik Sumskogo gosudarstvennogo universiteta. Tekhnicheskie nauki* [Herald of Sumy State University. Technical sciences], 2003, no. 11 (57), pp. 35-42.

21. Kas'ianov V.N., Kas'ianova E.V. Metody i sredstva adaptivnoi gipermedia [Methods and tools of adaptive hypermedia]. *Vychislitel'nye tekhnologii* [Computational technologies], 2004, vol. 9, pt. 2, pp. 333-341.

22. Badmaev B.Ts. *Psikhologiia i metodika uskorennogo obucheniia* [Psychology and methods of accelerated learning]. Moscow, Vlados, 1998. 272 p.

23. Gal'perin P.Ia. Psikhologiia myshleniia i uchenie o poetapnom formirovanii umstvennykh deistvii [Psychology of thinking and teaching about the gradual formation of mental actions]. *Issledovaniia myshleniia v sovetskoi psikhologii: sb. nauch. trudov* [Research of thinking in Soviet psychology: collect. of scientific works]. Moscow, Nauka, 1966, pp. 236-278.

24. Leont'ev A.N. *Izbrannye psikhologicheskie proizvedeniia: V 2 t.* [Selected psychological works. In 2 vols.]. Moscow, Pedagogika, 1983, vol. 1. 391 p.; vol. 2. 318 p.

25. Mashbits E.I. *Psikhologo-pedagogicheskie problemy komp'iuterizatsii obucheniia* [Psycho-pedagogical problems computerization of the learning]. Moscow, Pedagogika, 1988. 192 p.

26. Talyzina N.F. *Upravlenie protsessom usvoeniia znanii* [Management of the process of mastering of knowledge]. Moscow, MSU Publ., 1984. 288 p.

27. El'konin D.B. *Izbrannye psikhologicheskie Trudy* [Selected psychological works]. Moscow, Pedagogika, 1989. 560 p.

28. Jacobson A., Booch G., Rumbaugh J. *The Unified Software Development Process*. Addison-Wesley Publishing Co., 1999. 512 p. (Russ. ed.: Iakobson A., Buch G., Rambo Dzh. *Unifitsirovannyi protsess razrabotki programmnogo obespecheniial*. St. Petersburg, Piter, 2002. 496 p.).

29. Goldstein I.P. The Genetic Graph: a representation for the evolution of procedural knowledge. *International Journal of Man-Machine Studies*, 1979, no. 11, pp. 51-77.

30. Karpova I.P. *Issledovanie i razrabotka podsistemy kontrolia znanii v raspredelennykh avtomatizirovannykh obuchaiushchikh sistemakh. Kand. diss*. [Research and development of subsystem of the control of knowledge in distributed automated learning systems. Cand. diss.]. Moscow, 2002. 200 p.