

Методика разработки практических задач для автоматизированного контроля знаний и навыков при обучении ИТ-специалистов

И.С. Полевщиков^{1,2}

¹*Пермский национальный исследовательский политехнический университет*

²*Московский государственный университет пищевых производств*

Аннотация: В статье рассмотрены особенности предлагаемой методики разработки практических задач для автоматизированного контроля знаний и навыков при обучении ИТ-специалистов. Методика представлена на примере задач для контроля знаний и начальных навыков в области тестирования программного обеспечения (ПО) и описания функциональных требований к ПО. Применение методики способствует развитию средств электронного обучения и дистанционных образовательных технологий при подготовке ИТ-специалистов (в частности, студентов вузов по соответствующим направлениям).

Работа выполнена при поддержке стипендии Президента РФ молодым ученым и аспирантам (№ стипендии СП-100.2018.5), назначенной Советом по грантам Президента Российской Федерации.

Ключевые слова: ИТ-специалист, профессиональные знания и навыки, контроль знаний и навыков, автоматизированные обучающие системы, тестирование ПО.

В условиях развития средств электронного обучения и дистанционных образовательных технологий, актуальной является задача автоматизации контроля формирования компетенций у будущих ИТ-специалистов в ходе их подготовки [1-3]. Разработке и применению подобных средств обучения посвящены труды многих российских [4-6] и зарубежных авторов [7-9].

К настоящему времени реализован прототип автоматизированной системы управления (АСУ) формированием компетенций при подготовке ИТ-специалистов и частично внедрен на кафедре ИТАС ПНИПУ при обучении бакалавров направления «Программная инженерия».


Реализованный прототип АСУ позволяет студентам дистанционно отправлять отчеты по практическим и лабораторным работам на проверку. Преподаватель проверяет работу в соответствии с установленным ранее набором критериев, по каждому из них выставляет оценку и может указать замечания. Автоматически вычисляется комплексная оценка работы, которую можно откорректировать. Также допустимо указать замечания по

работе в целом, не привязывая их к конкретным критериям оценки.

Если студенту не понятны замечания или он не согласен с ними, то по любому из критериев оценки или по работе в целом студент может добавить свои примечания и отправить преподавателю. Тем самым обеспечена возможность обратной связи от студентов.

На рис. 1 представлен фрагмент веб-интерфейса АСУ с результатами проверки преподавателем лабораторной работы по теме «Тестирование базового пути» (в рамках дисциплины «Тестирование программного обеспечения» при подготовке бакалавров по направлению «Программная инженерия»). Красным цветом отмечены замечания преподавателя, сиреневым – примечания студента.

Проверка от 08.09.2020, 10:29 (степень выполнения работы: 63%):

Файл работы:  V1_Отчет_1_Иванов.pdf

Критерий	Оценка	Замечание/примечание
1. Задача №1	75	Нет замечаний Добавить примечание
1.1. Текст программы	100	Нет замечаний Добавить примечание
1.2. Нумерация операторов	100	Нет замечаний Добавить примечание
1.3. Поточковый граф	100	Нет замечаний Добавить примечание
1.4. Цикломатическая сложность V(G)	75	Нет замечаний Добавить примечание
1.5. Множество путей	50	См. пример, который разбирали на доске. Там я объяснял, как строить пути! Примечания студента: Я руководствовался примером про вычисление функции, который разбирали на паре 2 сентября, но не совсем понял ... X Добавить примечание
1.6. Тест-кейсы (исх. данные и ож. результаты)	60	Нет замечаний Добавить примечание
1.7. Тест-кейсы (сравнение реальных и ож. результатов)	50	Нет замечаний Добавить примечание

Рис. 1. – Пример веб-интерфейса с результатами проверки работы

АСУ позволяет работать с электронным архивом отчетов студентов, содержащим историю проверки каждого отчета (в случае наличия замечаний, после их исправления можно отправить отчет на проверку повторно).

Недостатком текущего прототипа системы является возможность преподавателя проверять практические задачи только вручную. Например, согласно рис. 1 критерий «1.3. Поточковый граф» предполагает ручную проверку преподавателем графа, построенного студентом в отчете по лабораторной работе. Однако для ускорения проверки начальных навыков построения потоковых графов требуется организовать генерацию задач на их построение и проверку автоматически.

Поэтому с целью дальнейшего развития АСУ предложена методика разработки практических задач при автоматизированном контроле знаний и навыков в процессе подготовки ИТ-специалистов (на примере задач на построение различных визуальных моделей, используемых для описания ПО). Особенности автоматической оценки подобных задач с применением теоретико-множественного подхода были показаны в работах [3, 10].

Методика основана на возможности генерации и оценки небольших практических задач для контроля знаний и навыков обучаемых в соответствии с шаблонами, составляемыми преподавателем. Особенности данной методики рассмотрим далее на примерах задач.

Задача №1. Наиболее простые задачи по теме «Тестирование базового пути» предназначены для проверки знаний и навыков расчета основных метрик, связанных с потоковым графом: число узлов, дуг, предикатных узлов, регионов, цикломатической сложности. Шаблон такой задачи включает в себя условия, связанные с данными метриками. Пример подобного шаблона:

- потоковый граф содержит 7-9 узлов;
 - потоковый граф содержит 2-3 предикатных узла;
-

– программа, соответствующая графу, содержит максимум один цикл.

Примеры трех конкретных потоковых графов в соответствии с данным шаблоном показаны на рис. 2-4.

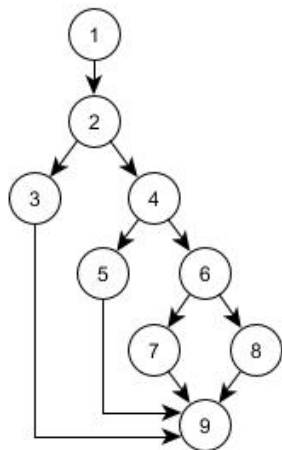


Рис. 2. – Пример №1
графа по шаблону

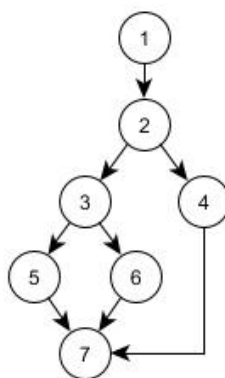


Рис. 3. – Пример №2
графа по шаблону

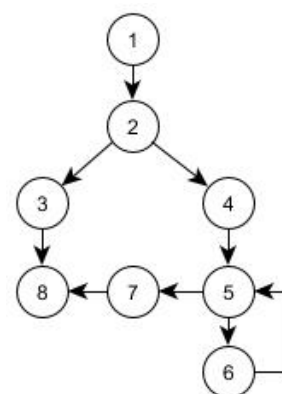


Рис. 4. – Пример №3
графа по шаблону

Для каждого обучаемого будет сформирован индивидуальный вариант задачи (по уровню сложности соответствующий рис. 2-4): на основе построенного автоматически графа студенту необходимо указать значения указанных выше метрик.

Задача №2. Более сложные задачи по теме «Тестирование базового пути» предполагают построение обучаемым потокового графа на основе сформированного автоматически (согласно шаблону) и пронумерованного кода программы на языке Java. Такие шаблоны основаны на описании структуры программы, в частности, указания наличия или отсутствия операторов ввода, вывода, выбора, циклов, простых или составных условий. Пример подобного шаблона:

- программа содержит любой вид цикла (`while`, `do-while` или `for`);
- операторы выбора и вложенные циклы отсутствуют;
- цикл содержит простое условие;
- программа содержит операторы ввода данных;
- программа содержит операторы вывода данных.

В таблице 1 представлены примеры трех конкретных программ на языке Java в соответствии с данным шаблоном.

Таблица №1

Примеры трех программ на Java в соответствии с шаблоном (задача №2)

№ примера	Код программы на Java	Особенности кода программы
1	<pre>int y=sc.nextInt(); // 1 int z=sc.nextInt(); // 1 while (z>=500){ // 2 z-=5*y; // 3 y--; // 3 } // 3 System.out.println("y="+y); // 4 System.out.println("z="+z); // 4</pre>	Программа содержит цикл с предусловием (while).
2	<pre>double x=sc.nextDouble(); // 1 int N=sc.nextInt(); // 1 for (int j=N /*2*/; j>15 /*3*/; j-- /*5*/) x+=5.75*j-1; // 4 System.out.println("x="+x); // 6</pre>	Программа содержит цикл с параметром (for).
3	<pre>int x1=sc.nextInt(); // 1 int x2=sc.nextInt(); // 1 do // 2 x1+=2*x2; // 2 while (x1<=175); // 3 System.out.println("x1="+x1); // 4</pre>	Программа содержит цикл с постусловием (do-while).

Задача №3. Изменим шаблон из задачи №2:

- программа содержит оператор выбора if;
- циклы и вложенные операторы выбора отсутствуют;
- оператор if содержит составное условие, состоящее из 2-х простых;
- программа содержит операторы ввода данных;
- программа содержит операторы вывода данных.

В таблице 2 представлены примеры трех конкретных программ на языке Java в соответствии с данным шаблоном.

Задача №4. Изменим шаблон из примеров №2, №3 (в результате программа усложнится с точки зрения построения потокового графа):

- программа содержит один оператор выбора if;

- программа содержит один цикл (while или do-while);
- все условия операторов if, while, do-while являются простыми;
- оператор if вложен в цикл или цикл вложен в оператор if;
- программа содержит операторы ввода данных;
- программа содержит операторы вывода данных.

Таблица №2

Примеры трех программ на Java в соответствии с шаблоном (задача №3)

№ примера	Код программы на Java	Особенности кода программы
1	<pre>int y1=sc.nextInt(); // 1 int y2=sc.nextInt(); // 1 if (y1<60 /*2*/ y2==300 /*3*/) y2*=3; // 4 else { // 5 y1-=10; // 5 y2%=2; // 5 } // 5 System.out.println("y1="+y1); // 6 System.out.println("y2="+y2); // 6</pre>	Условие содержит операцию дизъюнкции.
2	<pre>int x=sc.nextInt(); // 1 int y=sc.nextInt(); // 1 if (y>2*x /*2*/ && x>=25 /*3*/) { x+=y; // 4 --y; // 4 } // 4 else // 5 y*=2*x; // 5 System.out.println("x="+x); // 6 System.out.println("y="+y); // 6</pre>	Условие содержит операцию конъюнкции.
3	<pre>int w1=sc.nextInt(); // 1 int w2=sc.nextInt(); // 1 if (w1>=w2 /*2*/ w2<=7 /*3*/) w1=w2/2; // 4 else // 5 w2=w1+9; // 5 System.out.println("w1="+w1); // 6 System.out.println("w2="+w2); // 6</pre>	Условие содержит операцию дизъюнкции, но в отличие от примера программы №1 в данной таблице, включает другие переменные, другие операции отношений, и ветвь else содержит один оператор.

В таблице 3 представлены примеры трех конкретных программ на языке Java в соответствии с данным шаблоном.

Таблица №3

Примеры трех программ на Java в соответствии с шаблоном (задача №4)

№ примера	Код программы на Java	Особенности кода программы
1	<pre>int a=sc.nextInt(); // 1 int b=sc.nextInt(); // 1 if (a>=100){ // 2 while (b<50){ // 3 a-=3; // 4 b+=7; // 4 } // 4 } // 5 System.out.println("a="+a); // 6 System.out.println("b="+b); // 6</pre>	В оператор выбора if вложен цикл с предусловием while.
2	<pre>int a=sc.nextInt(); // 1 int b=sc.nextInt(); // 1 while (b>200){ // 2 if (a!=15) // 3 a--; // 4 b-=25; // 5 a+=4; // 5 } // 5 System.out.println("a="+a); // 6 System.out.println("b="+b); // 6</pre>	В цикл с предусловием while вложен оператор выбора if.
3	<pre>int a=sc.nextInt(); // 1 int b=sc.nextInt(); // 1 do { // 2 if (b==20){ // 3 a/=3; // 4 b=a+10; // 4 } // 4 b++; // 5 a--; // 5 } while (2*a>b); // 6 System.out.println("a="+a); // 7 System.out.println("b="+b); // 7</pre>	В цикл с постусловием do-while вложен оператор выбора if.

Дальнейшее усложнение задач на построение потокового графа может быть основано как на усложнении шаблона, так и на добавлении новых подзадач в соответствии с методом тестирования базового пути (например, помимо построения потокового графа, определить его цикломатическую сложность, определить базовое множество независимых линейных путей).

Задача №5. На основе шаблонов возможно разрабатывать задачи,

направленные на описание функциональных требований к ПО [3] посредством построения диаграмм Use Case UML. Шаблоны таких задач учитывают число актеров и вариантов использования на диаграмме, возможные отношения между данными элементами. Пример шаблона:

- диаграмма включает 2-3 актеров;
- диаграмма включает 5-6 вариантов использования;
- обязательно наличие отношения обобщения между актерами;
- обязательно наличие одного из двух видов отношений между вариантами использования: расширение или включение.

Можно предложить два вида задач. Первый вид предполагает построение диаграммы по предоставленному обучаемому текстовому описанию. При использовании второго вида обучаемому требуется ответить на список автоматически сгенерированных (иногда взаимосвязанных) тестовых вопросов по предоставленной диаграмме. Примеры вопросов: «Какой вид отношения между вариантами использования *B1* и *B2*?», «Какие варианты использования на данной диаграмме являются включаемыми?».

В таблице 4 представлены примеры трех конкретных диаграмм Use Case в соответствии с данным шаблоном.

Обобщая вышесказанное, схематично процесс контроля практических знаний и навыков студента (будущего ИТ-специалиста) на основе описанной методики и с учетом алгоритмов оценки задач (описанных в работах [3, 10]) показан на рис. 5. Преимуществом такого подхода является возможность автоматической генерации и оценки индивидуального варианта задачи для каждого студента.

Перспективами дальнейших исследований является применение описанной методики для разработки задач по различным видам знаний и навыков будущего ИТ-специалиста в соответствии с профессиональными и образовательными стандартами.

Таблица №4

Примеры трех диаграмм Use Case в соответствии с шаблоном (задача №5)

№ примера	Диаграмма Use Case UML	Описание диаграммы Use Case UML
1		<ul style="list-style-type: none"> – Актер <i>A1</i> инициирует выполнение варианта использования <i>B1</i>. – Актер <i>A2</i> инициирует выполнение вариантов использования <i>B2</i> и <i>B3</i>. – Актер <i>A3</i> инициирует выполнение варианта использования <i>B4</i>. – Актер <i>A1</i> является родителем актеров <i>A2</i> и <i>A3</i>. – Вариант использования <i>B5</i> является включаемым по отношению к варианту использования <i>B4</i>.
2		<ul style="list-style-type: none"> – Актер <i>A1</i> инициирует выполнение вариантов использования <i>B1</i> и <i>B2</i>. – Актер <i>A2</i> инициирует выполнение варианта использования <i>B3</i>. – Актер <i>A3</i> инициирует выполнение варианта использования <i>B4</i>. – Актер <i>A1</i> является родителем актера <i>A3</i>. – Вариант использования <i>B5</i> является расширяющим по отношению к варианту использования <i>B3</i>.
3		<ul style="list-style-type: none"> – Актер <i>A1</i> инициирует выполнение варианта использования <i>B5</i>. – Актер <i>A2</i> инициирует выполнение варианта использования <i>B6</i>. – Актер <i>A3</i> инициирует выполнение вариантов использования <i>B1</i>, <i>B2</i> и <i>B3</i>. – Актер <i>A3</i> является родителем актеров <i>A1</i> и <i>A2</i>. – Вариант использования <i>B4</i> является расширяющим по отношению к варианту использования <i>B2</i>.

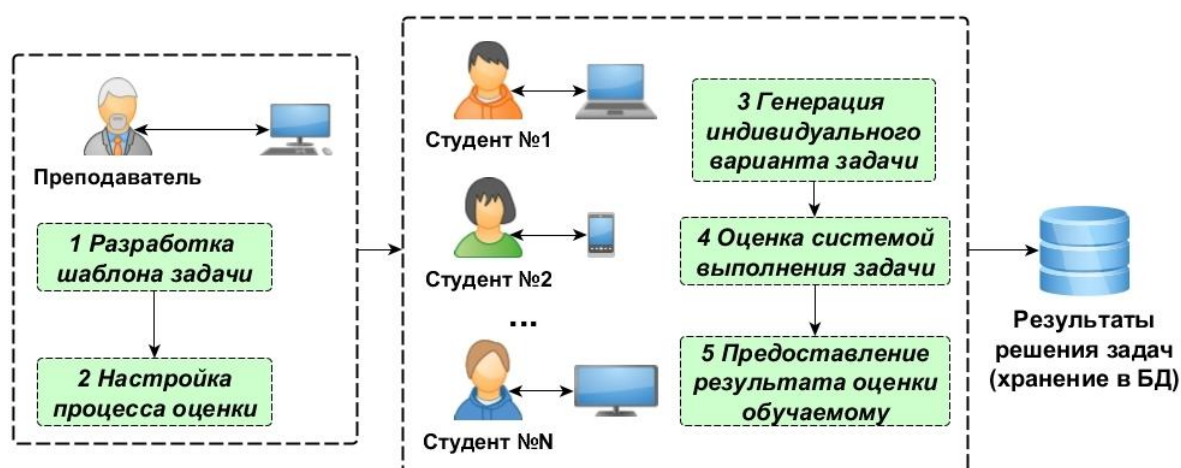


Рис. 5. – Контроль практических знаний и навыков студента

Литература

1. Леванов Д.Н. Модели адаптивного управления изложением материалов в электронных курсах для дистанционного обучения // Инженерный вестник Дона. 2013. №4. URL: ivdon.ru/ru/magazine/archive/n4y2013/2208.

2. Назаров А.И. Совершенствование электронного документооборота кафедры биомедицинского профиля на базе LMS Moodle // Инженерный вестник Дона. 2012. №4-1. URL: ivdon.ru/ru/magazine/archive/n4p1y2012/1223.

3. Полевщиков И.С. Применение средств автоматизации для совершенствования процессов контроля знаний и навыков при подготовке ИТ-специалистов // Информатизация образования и методика электронного обучения: цифровые технологии в образовании: материалы IV Междунар. науч. конф. Красноярск, 6–9 октября 2020 г.: в 2 ч. Ч. 1 / под общ. ред. Носкова М. В. – Красноярск: Сиб. федер. ун-т, 2020. – С. 308-312.

4. Зорин Ю.А. Автоматизация построения многовариантных тестовых заданий на основе деревьев и/или: дис. ... канд. техн. наук: 05.13.06. Санкт-Петербург, 2015. 140 с.

5. Говоров А.И., Говорова М.М., Слизень Е.В., Валитова Ю.О., Иванов

С.Е. Метод построения индивидуального образовательного маршрута при прохождении заданий на тренировку профессиональных умений по составлению SQL-запросов // Компьютерные инструменты в образовании. 2018. № 4. С. 45-62.

6. Романов Е.Л. Автоматизация учета рейтинга успеваемости студентов // Открытое и дистанционное образование. 2014. № 2 (54). С. 55-62.

7. Dan Bouhnik, Golan Carmi. E-learning Environments in Academy: Technology, Pedagogy and Thinking Dispositions // Journal of Information Technology Education: Research. 2012. V. 11. pp. 201-219.

8. Kovacic Z., Green J. Automatic Grading of Spreadsheet and Database Skills // Journal of Information Technology Education: Innovations in Practice. 2012. V. 11. pp. 53-70.

9. Gero A., Stav Y., Wertheim I., Epstein A. Two-tier multiple-choice questions as a means of increasing discrimination: case-study of a basic electric circuits course // Global Journal of Engineering Education. 2019. V. 21. №2. pp. 139-144.

10. Fayzrakhmanov R.A., Polevshchikov I.S. The Use of Mathematical Methods to Automate the Control of Skills in the Study of software Testing Algorithms // Proceedings of 2019 XXII International Conference on Soft Computing and Measurements (SCM): St. Petersburg, Russia, May 23–25, 2019 / IEEE Russia North West Section, Saint Petersburg Electrotechnical Univ. «LETI» (ETU «LETI»). [S. 1.]: IEEE, 2019. pp. 107-110.

References

1. Levanov D.N. Inzhenernyj vestnik Dona, 2013, №4. URL: ivdon.ru/ru/magazine/archive/n4y2013/2208.

2. Nazarov A.I. Inzhenernyj vestnik Dona, 2012, №4-1. URL: ivdon.ru/ru/magazine/archive/n4p1y2012/1223.

3. Polevshchikov I.S. Primenenie sredstv avtomatizatsii dlya sovershenstvovaniya protsessov kontrolya znaniy i navykov pri podgotovke IT-spetsialistov [Application of automation tools to improve knowledge and skills control processes in the training of IT specialists]. Informatizatsiya obrazovaniya i metodika elektronnoy obucheniya: tsifrovye tekhnologii v obrazovanii: materialy IV Mezhdunar. nauch. konf. Krasnoyarsk, 6–9 oktyabrya 2020 g.: v 2 ch. Ch. 1. Pod obshch. red. Noskova M. V. Krasnoyarsk: Sib. feder. un-t, 2020. pp. 308-312.
 4. Zorin Yu.A. Avtomatizatsiya postroeniya mnogovariantnykh testovykh zadaniy na osnove derev'ev i/ili: dis. ... kand. tekhn. nauk [The thesis of Candidate of Technical Sciences]: 05.13.06. Sankt-Peterburg, 2015. 140 p.
 5. Govorov A.I., Govorova M.M., Slizen' E.V., Valitova Yu.O., Ivanov S.E. Komp'yuternye instrumenty v obrazovanii. 2018. № 4. pp. 45-62.
 6. Romanov E.L. Otkrytoe i distantsionnoe obrazovanie. 2014. № 2 (54). pp. 55-62.
 7. Dan Bouhnik, Golan Carmi. Journal of Information Technology Education: Research. 2012. V. 11. pp. 201-219.
 8. Kovacic Z., Green J. Journal of Information Technology Education: Innovations in Practice. 2012. V. 11. pp. 53-70.
 9. Gero A., Stav Y., Wertheim I., Epstein A. Global Journal of Engineering Education. 2019. V. 21. №2. pp. 139-144.
 10. Fayzrakhmanov R.A., Polevshchikov I.S. The Use of Mathematical Methods to Automate the Control of Skills in the Study of software Testing Algorithms. Proceedings of 2019 XXII International Conference on Soft Computing and Measurements (SCM): St. Petersburg, Russia, May 23–25, 2019. IEEE Russia North West Section, Saint Petersburg Electrotechnical Univ. «LETI» (ETU «LETI»). [S. l.] : IEEE, 2019. pp. 107-110.
-