УДК 621.372

РАЗРАБОТКА МЕТОДА ВЕРИФИКАЦИИ ВИРТУАЛЬНЫХ ТРЕНАЖЁРОВ РАБОЧИХ МЕСТ

Н.Н. Войт[4], С.Ю. Кириллов[5], С.И. Бочков[6]

Виртуальные компьютерные промышленные тренажёры позволяют за меньшее по сравнению с традиционным методами обучения время подготовить персонал рабочих профессий. Вместе с тем большое значение имеет соответствие моделируемого рабочего места реальному месту, организованному в соответствии с технологическим процессом. Традиционные методы тестирования – ручное и автоматическое юнит-тестирование – зачастую не позволяют полностью покрыть функционал тренажёра и проверить его в кратчайшие сроки. В статье предложен метод верификации виртуальных тренажёров, основанный на структурно-параметрическом анализе его содержимого и установлении его соответствия с данными из PLM-систем.

Введение

Проектирование, реализация и внедрение виртуальных тренажёрных систем сборки промышленных изделий решает проблему подготовки и переподготовки персонала и специалистов в сокращённые сроки и повышение качества обучения без отрыва от производства.

Во время разработки виртуальных тренажёров [1, 2, 3] следует придерживаться технического задания, разработанного и согласованного с заказчиком. В свою очередь, оно основано на технологическом процессе, применяемом на реальном рабочем месте данной предметной области, а также нормативно-технической документации, на которую ссылается технологический процесс: спецификации изделия, государственным, отраслевым стандартам. Допускается добавлять, изменять или пропускать только те действия на рабочем месте, которые не нарушают логики технологического процесса (например, подсветка радиоэлемента, автоматическая настройка оборудования).

[4]*432027, Ульяновск, ул. Северный Венец, 32, УлГТУ, e-mail: n.voit@ulstu.ru*

[5]*432027, Ульяновск, ул. Северный Венец, 32, УлГТУ, e-mail: bochkovsi@ido.ulstu.ru*

[6]*432027, Ульяновск, ул. Северный Венец, 32, УлГТУ, e-mail: kirillovsyu@gmail.com*

Для проверки виртуального тренажёра на его адекватность реальному рабочему месту используются тестирование его экспертом, автоматические или полуавтоматические тесты (юнит-тесты, основанные на методе "чёрного ящика"). Однако, если тренажёр имеет большую функциональность, ручное тестирование и разработка юнит-тестов может занять достаточно большое количество времени, к тому же последние могут покрыть не все функциональные возможности тренажёра.

Для полного автоматического тестирования таких тренажёров предлагается метод верификации виртуального тренажёра рабочего места, основанный на структурно-параметрическом анализе содержимого тренажёра и установке его соответствия с внешним источником данных с помощью онтологии.

1. Современное состояние исследований

Верификацией программы называется деятельность, направленная на установление правильности или ошибочности выполнения программы; проверка соответствия программы или промежуточного результата проектирования предъявляемым к ней требованиям [14].

Кулямин В.В. приводит следующую классификацию методов верификации программного обеспечения (ПО) [5].

Экспертиза представляет собой анализ ПО, проводимый экспертом. Данный подход имеет высокую функциональную пригодность и способен решать огромный круг задач тестирования ПО, при этом может быть применим к любым его свойствам на любом этапе тестирование программ. В то же время экспертиза не может быть автоматизирована и требует активного участия людей.

Статический анализ используется для проверки формализованных правил корректного построения артефактов ПО и поиска часто встречающихся дефектов по некоторым шаблонам. Такой анализ хорошо автоматизируется и может быть практически полностью возложен на инструменты, не требующие специальной подготовки. Большинство техник статической проверки, доказавших свою эффективность на практике, становятся частью компиляторов или даже преобразуются в семантические правила языков программирования. Статический анализ наиболее рационально применим на этапе создания ПО, так как он пригоден только к фрагментам программы и не требует их исполнения, что позволяет уменьшить стоимость проекта и увеличить его надёжность.

Формальные методы используют для анализа свойств ПО формальные модели требований, поведения ПО и его окружения. Анализ формальных моделей выполняется с помощью специфических техник, таких как дедуктивный анализ, проверка моделей или абстрактная интерпретация.

Формальные методы применимы только к свойствам и артефактам, формально выраженным в рамках некоторой математической модели.

В рамках *динамических методов* анализ и оценка свойств ПО делаются по результатам реальной работы ПО или некоторых его моделей. Примерами такого рода методов являются обычное тестирование или имитационное тестирование, мониторинг, профилирование. Для применения динамических методов необходимо иметь работающую систему или хотя бы некоторые её компоненты, или же их прототипы.

Существуют также *синтетические методы*, в рамках которых применяются элементы нескольких перечисленных выше видов верификации. В отдельные области выделены динамические методы, использующие элементы формальных, — тестирование на основе моделей [73] и мониторинг формальных свойств. Общей идеей таких методов является сочетание преимуществ основных подходов к верификации, сводя к минимуму их недостатки.

Существует множество средств формальной верификации программ [6].В работах [7, 8] представлен плагин CONC2SEQ, написанный для платформы FRAMA-C и верификации параллельных программ на языке C. CONC2SEQ переводит исходный параллельный код в последовательный для корректной работы в FRAMA-C, при этом пользовательские спецификации автоматически интегрируются в новый код. Инструмент Lazy-Cseq [9, 10] позволяет перевести многопоточную программу на языке C в недетерминированный последовательный код, сохраняющий достижимость для всех циклических трассировок с заданным числом итераций. Он повторно использует существующие высокопроизводительные инструменты ограниченной проверки моделей (bounded model checking, BMC). Проект Java Pathfinder, изначально создававшийся как средство проверки модели с явным состоянием для байт-кода Java [11], способен выполнять проверку модели программ, написанных на языках, ориентированных на виртуальную машину Java. Первоначальный подход к реализации заключался в преобразовании байт-кода Java в Promela для анализа с помощью средства проверки модели Spin [12]. На данный момент Java Pathfinder проверяет байт-код Java напрямую и применяется для анализа параллельных программ и абстрактных моделей, в том числе в приложения на платформе Android [13].

2. Метод верификации

В логико-алгебраическом смысле модель виртуального рабочего места имеет следующий вид [8]:

Тренажёр рабочего места = (множество объектов сборки; множество расходных материалов; множество инструментов; множество документов; целевой объект; технологический процесс),

где:

Объект сборки = (множество пар «параметр-значение»)

Расходный материал = (множество пар «параметр-значение»)

Документ = (наименование; объект сборки или расходный материал; множество пар «параметр-значение»)

Инструмент = (множество пар «параметр-значение»)

Целевой объект – состояние объекта (объектов) сборки, характеризующее завершённость сборки того или иного изделия.

Технологический процесс = (основные операции, промежуточные операции), где:

Основная операция: объекты сборки × материалы × инструменты × документы → объекты сборки × материалы × инструменты – операция, меняющая состояние одного или нескольких объектов сборки.

Промежуточные операции = (активация/деактивация инструмента; активация/деактивация объекта; активация/деактивация материала), где

Активация/деактивация инструмента: инструмент → инструмент – взятие, смена инструмента

Активация/деактивация объекта: объект сборки → объект сборки – взятие, смена объекта сборки

Активация/деактивация материала: расходный материал → расходный материал – взятие, смена расходного материала

Источником данных для тестирования является система управления жизненным циклом изделия (PLM) [4]. На рис. 1 представлена организация данных в PLMв форме классифицирующей онтологии.

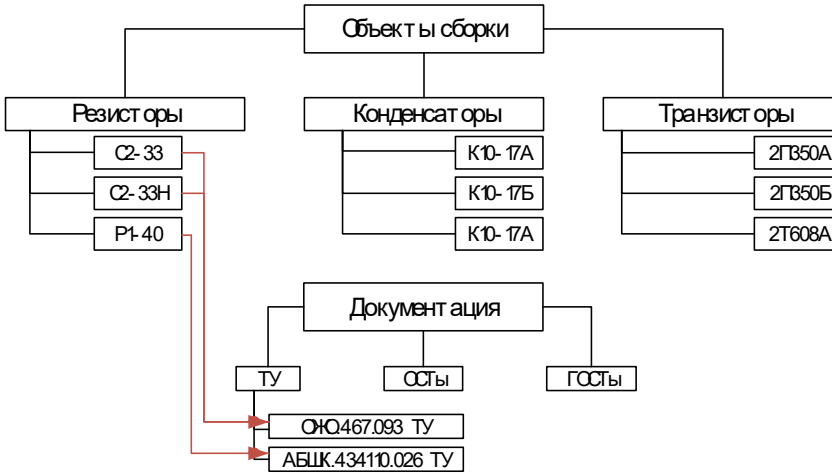

Рис. 1. Онтологическая модель данных в PLM

На рис. 2 представлены диаграммы, поясняющие работу метода структурно-параметрического тестирования виртуального рабочего места (ВРМ).

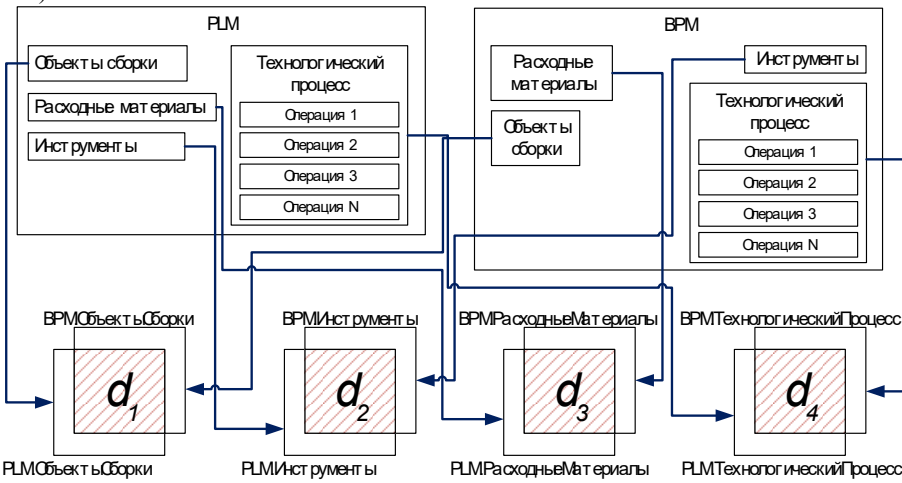

Рис. 2. Структурная схема процесса тестирования ВРМ

Процесс тестирования включает в себя следующие подпроцессы:

- соответствие объектов сборки в ВРМ объектам сборки, указанным в спецификации ($d_1 = \frac{AO_V}{AO_{PLM}}$, где AO_V – количество объектов сборки в ВРМ, указанных в спецификации в PLM,

AO_{PLM} – количество объектов сборки, указанных в спецификации в PLM);

- соответствие инструментов в BPM инструментам, указанным в спецификации ($d_2 = \frac{T_V}{T_{PLM}}$, где T_V – количество инструментов в BPM, указанных в спецификации в PLM, T_{PLM} – количество инструментов, указанных в спецификации в PLM));
- соответствие расходных материалов в BPM расходным материалам, указанным в спецификации ($d_3 = \frac{CM_V}{CM_{PLM}}$, где CM_V – количество расходных материалов сборки в BPM, указанных в спецификации в PLM, CM_{PLM} – количество расходных материалов, указанных в спецификации в PLM);
- соответствие содержания и порядка следования операций технологического процесса в BPM техпроцессу, хранящемуся в PLM ($d_4 = \frac{O_V}{O_{PLM}}$, где O_V – количество шагов техпроцесса, заложенного в BPM, которые, начиная с первого, соответствуют шагам техпроцесса, указанного в документации в PLM, O_{PLM} – количество шагов техпроцесса, заложенного в документации в PLM).

Результатом тестирования BPM является величина соответствия D, равная наименьшему из результатов выполнения подпроцессов, т.е. $D = \min(d_1, d_2, d_3, d_4)$.

Заключение

В условиях невозможности проведения ручного экспертного тестирования виртуального компьютерного тренажёра или автоматического юнит-тестирования, покрывающего все функциональные возможности тренажёра, находят применение новые методы тестирования, основанные на источниках, регламентирующих реальный рабочий процесс.

Для данного метода предложена модель виртуального тренажёра рабочего места, интенсивно использующая понятия "объект сборки", "инструмент", "расходный материал".

Разработанный метод является универсальным и подходит для виртуальных рабочих мест независимо от предметной области рабочей профессии.

Исследование выполнено при финансовой поддержке РФФИ в рамках научного проекта № 17-07-01417. Исследование выполнено при финансовой поддержке РФФИ и Правительства Ульяновской области в рамках научного проекта № 18-47-730032. Исследования поддержаны

грантом Министерства образования и науки Российской Федерации, проект № 2.1615.2017/4.6.

Список литературы

1. Afanasyev, A., Afanasyeva, T., Bochkov, S., & Voit, N. (2018). Application of virtual reality technology in the learning process. Proceedings of 10th annual International Conference on Education and New Learning Technologies (EDULEARN18). https://doi:10.21125/edulearn.2018.2485
2. Бочков С.И. Разработка виртуальных рабочих мест радиотехнических специальностей на примере объёмного монтажа. / Сборник научных трудов X Всероссийской школы-семинара аспирантов, студентов и молодых ученых "Информатика, моделирование, автоматизация проектирования" (ИМАП-2018), с. 67-78.
3. Войт Н.Н., Канев Д.С., Бочков С.И., Уханова М.Е. Разработка программного обеспечения оценки действий обучаемых в виртуальном окружении на основе экспертной системы / Электронное обучение в непрерывном образовании 2018. V Международная научно-практическая конференция. – 2018. – С. 151-159.
4. Уханова М.Е. Разработка семантической модели организационно-технических компонентов конструкторского проектирования на основе онтологии / Информационно-измерительные и управляющие системы" – №1(11). – С. 98-108. https://doi.org/10.18127/j20700814-201811-16
5. Кулямин В.В. Методы верификации программного обеспечения. – М.: ИСП РАН, 2008. – 111 с.
6. Survey of Existing Tools for Formal Verification. Robert C. Armstrong, Ratish J. Punnoose, Matthew H. Wong, Jackson R. Mayo https://prod-ng.sandia.gov/techlib-noauth/access-control.cgi/2014/1420533.pdf
7. Frama-C: https://frama-c.com/index.html
8. Blanchard A., Kosmatov N., Lemerre M. & Loulergue F., “Conc2Seq: A Frama-C Plugin for Verification of Parallel Compositions of C Programs”, Source Code Analysis and Manipulation (SCAM), (2016), 767-772.
9. Tomasco E., Nguyen T.L., Inverso O., Fischer B., La Torre S. & Parlato G. (2016) “Lazy sequentialization for TSO and PSO via shared memory abstractions”, Proceedings of the 16th Conference on Formal Methods in Computer-Aided Design. FMCAD Inc, pp.193-200.
10. C Sequentialisation Tool for Software Verification / https://www.southampton.ac.uk/~gp1y10/cseq/
11. Havelund K., & Pressburger T. (2000) “Model checking JAVA programs using JAVA PathFinder”, International Journal on Software Tools for Technology Transfer, vol. 2 (4), pp. 366-381. https://doi.org/10.1007/s100090050043
12. K. Havelund. (1999) Java PathFinder: A Translator from Java to Promela, in Proceedings of the 5th and 6th International SPIN Workshops on Theoretical and Practical Aspects of SPIN Model Checking, (London, UK, UK), pp. 152–, Springer-Verlag, 1999. https://doi.org/10.1007/3-540-48234-2_11

13. Kohan A., Mitsuharu Y, Artho C., Yoriyuki Y., Lei M., Masami H., & Yoshinori T. (2017) Java Pathfinder on Android Devices, ACM SIGSOFT Software Engineering Notes, 41 (6), pp. 1-5
https://doi.org:10.1145/3011286.3011292
14. Камкин А.С. Введение в формальные методы верификации программ: учебное пособие. – М.: МАКС Пресс, 2018. – 272 с.
15. Voit N., Bochkov S., Kirillov S.Virtual workplaces testing method on accordance with the technical task, in 4th International Conference on Computational Intelligence & Communication Technology (CICT), 2019 (preprint).