

УДК 004.896

В.А. Лазарев

ПОСТРОЕНИЕ ИНТЕЛЛЕКТУАЛЬНОЙ СИСТЕМЫ СОПРОВОЖДЕНИЯ АВТОМАТИЧЕСКОГО ТЕСТИРОВАНИЯ

Нижегородский государственный технический университет им. Р.Е. Алексеева

Приведена методика применения продукционной модели представления знаний в системах поддержки систем автоматического тестирования. Приводится оценка выгоды от внедрения подобной системы на базе реального проекта.

Ключевые слова: продукционная система, автоматическое тестирование, интеллектуальная система сопровождения.

Введение

С развитием компьютерных технологий развивалось и производство программных средств. В настоящий момент производится сотни тысяч различных программных продуктов. Конкурентная борьба приводит к постоянному сокращению времени от начала разработки продукта до вывода продукта на рынок. Это приводит к сокращению всех этапов разработки программного обеспечения, в том числе и фазы тестирования как наиболее ответственной и трудоемкой. Для сохранения приемлемого качества продукта, стандартом стала автоматизация тестирования разрабатываемого продукта. В то же время значимость систем автоматической поддержки стремительно растет и, по оценкам экспертов, через несколько лет ее значимость будет соизмерима со значимостью САПР. В данной статье разрабатывается система поддержки автоматического тестирования с использованием продукционной модели представления знаний.

Анализ предметной области

Рассмотрим пример существующей системы автоматизации тестирования. Тестируемый продукт тесно взаимодействует с операционной системой и аппаратными средствами. Использование виртуальных машин невозможно из-за неадекватного эмулирования ими аппаратной части (используются новейшие и перспективные наборы команд, поддержка которых отсутствует в существующих виртуальных машинах). Продукт поддерживает более 50 различных программно-аппаратных платформ, включая:

- мобильные платформы: планшетные компьютеры и смартфоны под управлением Windows и Android;
- настольные компьютеры под управлением различных версий Windows и Linux;
- серверные платформы под управлением различных Linux систем.

В результате обсуждения были выбраны 35 платформ, на которых требуется тестировать продукт. Для покрытия всех платформ была сформирована тестовая лаборатория, содержащая 150 тестовых машин распределенных на трех, географически удаленных площадках (Россия, Израиль, США).

Тестирования продукта состоит из двух этапов:

- интеграционного тестирования – после каждого изменения в продукте набор коротких тестовых пакетов (общая длительность менее часа), на ограниченном наборе платформ (5). На этом этапе выявляются 80% регрессий;
- еженедельного системного тестирования, где выполняется полный набор тестов.

Рассматривается система, являющаяся вторым этапом тестирования.

Среди тестовых машин присутствуют инженерные образцы перспективных аппаратных платформ. С целью уменьшения влияния аппаратных сбоев на результаты тестирования используются только наиболее стабильные инженерные образцы, инженерные образцы выводятся из автоматического тестирования, как только появляется более надежная машина соответствующая данной платформе (табл. 1).

Таблица 1

Распределение тестовых машин по стабильности

Инженерные образцы А0 (заявленное время наработки на отказ 8 ч), %	10
Инженерные образцы В0 (заявленное время наработки на отказ 24 ч), %	20
Официально выпущенные системы (заявление время наработки на отказ 14 дней), %	70

При этом так как платформы группируются, в том числе и по аппаратному признаку, ряд платформ состоит только из инженерных образцов.

Общее количество тестов около 50 000. Они сгруппированы в 120 тестовых пакетов. С учетом того, что один и тот же тест запускаются на различных платформах, запускается порядка трех тысяч тестовых пакетов на различных тестовых платформах. Время исполнения тестов различно и зависит как от самого теста, так и от платформы, на которой он исполняется. Время исполнения одного тестового пакета за 6 месяцев распределились следующим образом:

- 31,90% тестов делятся менее 1 мин (в большой степени это означает инфраструктурную проблему или отмену пользователем до начала исполнения);
- 28,67% тестов делятся менее 10 мин;
- 23,32% делятся от 10 мин до ч;
- 13,34% делятся от 1 ч до 8 ч;
- 2,29% делятся от 8 ч до суток;
- 0,45% тестов делятся от суток до трех.

Для исполнения всех тестов на некоторых платформах требуется более 6 суток. С учетом того, что требуется перезапуск отдельных тестов для анализа причин падения и того, что тесты прогоняются на еженедельной основе, ожидается постоянная работоспособность системы автоматизированного тестирования. А учитывая использование инженерных образцов, географическое распределение тестовых машин, требуется постоянное приложение усилий для поддержания ее в рабочем состоянии. При этом строгие временные рамки приводят значительной задержке в анализе проблем:

- тесты запускаются раз в неделю;
- нередко проблема анализируется только после повторного воспроизведения (задержка неделя);
- из-за занятости тестовых машин анализ откладывается еще на три-пять дней.

В результате нередко причина проблемы выясняется через три-четыре недели после ее внесения.

В процессе работы системы замечено, что около 18% от всего количества запусков тестов - это перезапуски по различным причинам. Далее (табл. 2) приведена статистика по перезапускам за 6 мес. в процентах от общего количества перезапусков, возможность провести перезапуск в автоматическом режиме и среднее время, затрачиваемое на анализ проблемы.

Как следует из таблицы, около 30% перезапусков могут быть обработаны в автоматическом режиме. При этом существенно сокращается время на анализ тестовых падений и перезапуск тестов (около 30% от времени затраченного на сопровождение тестирования).

Таблица 2

Причины перезапусков тестов

Причина перезапуска	%	авто	время
Ошибка подготовки тестового цикла	16,53	Нет	10 мин
Неверное окружение на тестовой машине	7,2	Да	1 час
Система непрерывной интеграции	0,68	Да	10 мин
Действий пользователя	3,61	Да	10 мин
Проблема с сетью	2,97	Нет	10 мин
Пользовательские действия по анализу падений	17,65	Нет	-
Проблема с доступом к сетевому хранилищу	5,78	Да	10 мин
Проблема зеркалирования тестовой инфраструктуры	5,46	Да	1 день
Перезапуск после исправления теста	33,11	Нет	-
Зависание теста или маленький таймаут	1,19	Да	10 мин
Нестабильная тестовая машина	5,8	Да	1 день

Для крупных проектов объем тестируемой функциональности приводит к росту количества и длительности тестов. Кроме того, большинство современных продуктов поддерживают различные платформы (программные и аппаратные), в результате необходимо запускать каждый тест на различных платформах. Для решения задач по распределению автоматических тестов на гетерогенном парке машин используются системы непрерывной интеграции (continuous integration systems). Примерами подобных систем могут служить Jenkins/Hudson, Pulse, Team City, Electric Commander и другие.

Рассмотрим типичные требования к системе непрерывной интеграции и их основные характеристики. Основным требованием является максимально быстрое и надежное предоставление тестовых результатов. Из этих требований следуют необходимость включения множества тестовых машин (повышение надежности за счет дублирования и скорости тестирования за счет параллелизма) и сетевой характер системы автоматизации тестирования. Нередко продукт является кроссплатформенным либо поддерживает различные версии одной ОС, в результате требуется гетерогенность тестовых платформ.

Помимо собственно процесса тестирования, такие системы обслуживают процесс построения продукта и тестов. Кроме того, существует подсистема контроля статуса всех машин, подключенных к системе. Чаще всего такие системы выполняют функции администрирования тестового парка. Кроме того, для сокращения издержек и максимизации использования существующих ресурсов от систем автоматизации тестирования требуется непрерывная работа 24 часа в день, 7 дней в неделю. Для поддержки подобной системы требуются знания в различных областях (знание языков программирования, навыки администрирования, понимание архитектуры продукта и тестовой системы и т.д.), а значит, необходим персонал высокой квалификации.

Анализ предметной области функционирования систем автоматического тестирования показал, что они высокого уровня сложности, гетерогенны по множеству выполняемых функций и поддерживаемых процессов, загружены 24 часа в сутки, 7 дней в неделю, требуют для поддержки и использования персонала высокой квалификации.

Таким образом, внедрение систем автоматизации тестирования программного обеспечения актуализирует комплекс связанных задач в плане создания методологии и инструментов, способных заменить высококвалифицированный персонал средствами интеллектуальной поддержки систем автоматизации процесса тестирования.

Предлагаемая методика

Для эффективного решения задачи планируется создать интеллектуальную систему, осуществляющую анализ текущего состояния процесса автоматического тестирования и коррекцию ошибочных состояний в максимально автоматическом режиме. Данные о состоянии тестов и машин поступают из базы данных, системы распределения задач и узлов в сети. После обработки система интеллектуальной поддержки вырабатывает корректирующие действия.

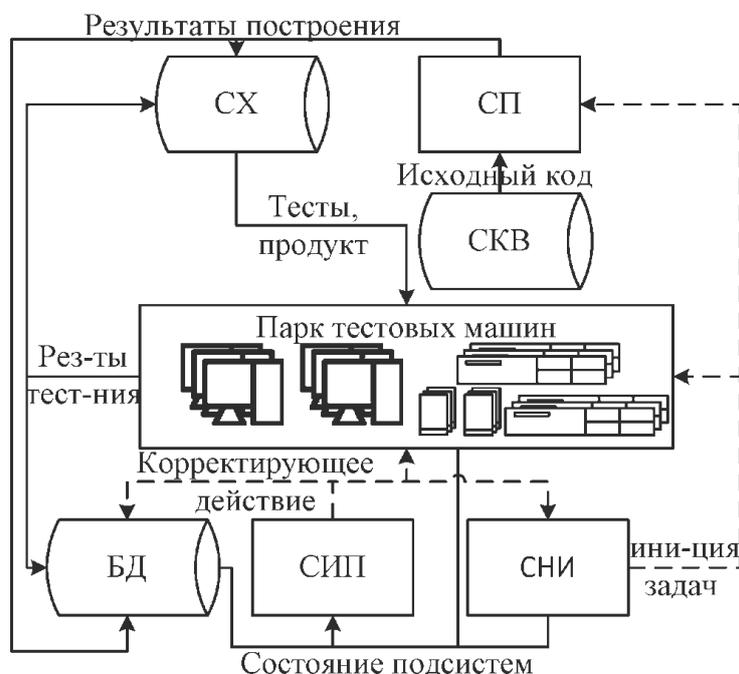


Рис. 1. Архитектура системы

На рис. 1 представлена общая архитектура системы автоматического тестирования, состоящей из базы данных (БД), системы непрерывной интеграции (СНИ), системы контроля версий исходного кода (СКВ), сетевого хранилища (СХ), парка тестовых машин, системы построения тестов и продукта (СП) и системы интеллектуальной поддержки процесса тестирования (СИП).

Интеллектуальную систему сопровождения автоматического тестирования планируется внедрять в существующую систему автоматического запуска тестов. За время использования существующей системы пользователи создали документ, описывающий типичные проблемы и способы реакции на них. Так как знания в СИП описывают реакции на различные состояния системы и рекомендации по переводу в желаемое состояние, пользователи привыкли к форме представления знаний ЕСЛИ-ТО, будет использована продукционная модель представления знаний.

В результате, интеллектуальная подсистема сопровождения процесса автоматического тестирования представляет собой экспертную систему (на базе оболочки CLIPS), использующую продукционную модель представления знаний. Правила создаются, корректируются, удаляются администратором системы на этапе первичного обучения и при выявлении несоответствий в процессе эксплуатации. Пользователь включается в процесс сопровождения автоматического тестирования, если:

- осуществляется первичный ввод данных (информация о новых/измененных тестах, добавление/удаление информации о тестовых машинах, изменение критериев качества и т. д.);
- действие невозможно выполнить системой (требуется физический доступ к машинам, требуется общение с соседними командами, проблемы в сети);

- появилось новое знание, требуется подтверждение пользователя;
- предлагаемое корректирующее воздействие слишком опасно;
- ни одно правило не как реальная и нужен анализ продукта и теста.

В работе системы сопровождения автоматического тестирования можно выделить ряд типовых задач, при этом каждой типовой задаче соответствует множество типовых сценариев. Далее (табл. 3) представлены типовые задачи, характерные для систем сопровождения автоматизированного тестирования программного обеспечения, классифицированные по мобильности (переносимости на системы автоматизированного тестирования, отличные от рассматриваемой) и применимости продукционной модели представления знаний.

Таблица 3

Типовые задачи

Типовая задача	Прод.	Моб.
Управление состоянием парка тестовых и служебных машин	+	+
Управление конфигурацией тестов	+	+
Управление запуском тестового пакета	+	+
Управление процессом исполнения тестов	+	+/-
Управление процессом сохранения тестовых результатов	+	+
Первичный анализ тестовых результатов и связь их с системой учета ошибок	+	-
Анализ статистики запусков и генерация рекомендаций по оптимизации тестирования	-	+
Генерация новых правил и знаний	+	+/-

Как видно из табл. 3, большинство типовых задач переносимы или частично переносимы, что позволит внедрять разрабатываемую систему на различных предприятиях. Каждая типовая задача поддерживается отдельным фрагментом базы знаний, что позволяет комбинировать набор типовых задач, которые требуется решать в рамках внедрения системы на различных площадках. Далее рассматривается фрагмент предметной области, описывающий одну типовую задачу и ее типовые сценарии.

При автоматическом тестировании необходимо поддерживать парк тестовых машин, разнородный по своему аппаратному и программному составу. Для контроля состояния парка тестовых машин происходит проверка на регулярной основе каждой из них, при этом существует несколько возможных сценариев:

- недоступна физически. В данном случае требуется вмешательство администратора лаборатории. Соответствующее уведомление отправляется ответственным сотрудникам;
- доступна физически, но недоступна для системы непрерывной интеграции. Необходимо проанализировать причину и в зависимости от нее можно предпринять корректирующие действия:
 - отключена пользователем в системе непрерывной интеграции. Необходимо уведомить отключившего пользователя и администратора. Пользователь или администратор включает машину по завершении всех работ;
 - отключена автоматически из-за проблемного состояния ресурсов машины. Требуется вмешательство администратора;
 - отключена автоматически по служебным причинам. Если задача, которая отключила машину, уже не исполняется, необходимо включить данную машину;
 - агент не доступен;
 - если машина сконфигурирована корректно – рестарт машины;
 - если ошибка конфигурации – уведомить администратора;

- доступна для системы распределения задач, но невозможно исполнение тестов. Необходимо анализировать причины падения тестов:
 - не хватает прав для исполнения теста – скорректировать политики безопасности;
 - возникает ошибка в подсистеме настройки окружения – необходимо отключить машину в системе распределения задач;
 - недоступны сетевые диски – подключить их;
 - при попытке инициировать исполнение очередной задачи возникает ошибка системы распределения задач – отключить машину в системе распределения задач;
- машина доступна и тесты исполняются, предоставляя достоверные результаты.

На рис. 2 представлено дерево принятия решений для подсистемы контроля и коррекции состояния машин.



Рис. 2. Дерево принятия решений

Далее содержится список корректирующих действий, соответствующих листьям дерева принятия решений:

1. Ничего.
2. Изменить настройки групповой политики.
3. Отключить машину и уведомить администратора.
4. Подключить сетевой диск.
5. Отключить машину и уведомить администратора.
6. Уведомить пользователя и администратора (повторять раз в сутки).
7. Отправить уведомление администратору.
8. Если зеркалирование завершилось, включить машину.
9. Если тест, требующий эксклюзивного исполнения, завершился, включить машину.
10. Отправить уведомление администратору.
11. Рестартовать машину.
12. Сконфигурировать агент автоматически.

Выводы

Внедрение системы интеллектуальной поддержки автоматического тестирования позволяет оптимизировать работу отдела занимающегося тестированием, при этом:

- сократится время реакции на проблемы в продукте;
- сократится время простоя тестовых машин;
- повысится качество тестирования за счет анализа больших объемов данных;
- снижается вероятность недостоверных результатов тестирования;
- сократится время затрачиваемое сотрудниками на рутинные действия и освободится время на более глубокий анализ проблем в продукте и развитие тестов.

В результате, используя те же ресурсы, можно будет более полно и качественно тестировать продукт. Как показал анализ типовых задач, использование данного подхода позволит переносить разработанную систему на различные площадки без существенных изменений в ядре системы.

Но, с другой стороны, внедрение данной системы делает актуальной задачу создания средств автоматизации выделения знаний о предметной области и их представлении в машинно-ориентированном виде.

Библиографический список

1. Джонс, М.Т. Программирование искусственного интеллекта в приложениях / М.Т. Джонс.– М.: ДМК Пресс, 2006. – 312 с.
2. Quinlan, J. R. Generating Production Rules From Decision Trees // Knowledge acquisition, 1987. P. 304–307.
3. Мисевич, П.В. Сценарно-ситуационный подход к проектированию средств интеллектуальной поддержки процесса функционирования автоматизированных систем // Системы управления и информационные технологии. 2007. N2.1(28). С. 166–171.

*Дата поступления
в редакцию 01.02.2014*

V.A. Lazarev

DEVELOPING OF INTELLECTUAL SUPPORT SYSTEM FOR AUTOMATED TESTING

Nizhny Novgorod state technical university n.a. R.E. Alexeev

Purpose: Optimizing resources needed for automatic validation process with increasing validation quality.

Design/Methodology/Approach: The paper proposes production rules system as intellectual core for test automation system. The subject area is analyzed basing on real project and one typical task is reviewed in terms of production rules system.

Findings The production rules system can be used as core of intellectual support system for automation testing.

Research limitations/implications: The system actualizes task of automatic selecting and converting into machine oriented form knowledge about subject area.

Originality/Value: The presented approach allows to reduce resources (including hardware and test engineer time) required for automated testing and improves quality of testing by deeper analysis of test failures.

Key words: production system, automated testing, intellectual support.