XXXIII Неделя науки СПбГПУ. Материалы межвузовской научно-технической конференции. Ч.V: С.28-29, 2005.

УДК 658.512.011.56: 681.3.06

П.Д.Дробинцев (асп., каф. ИУС), М.Ш.Даишев (5 курс, каф. ИУС),
Д.В.Песков (5 курс, каф. ИУС), Ю.В.Юсупов (5 курс, каф. ИУС),
В.П.Котляров, к.т.н., проф.

ПИЛОТИРОВАНИЕ ИНТЕГРИРОВАННОЙ ТЕХНОЛОГИИ ТЕСТИРОВАНИЯ И ВЕРИФИКАЦИИ

Современный уровень требований к качеству программного продукта хотя и обеспечивается в рамках традиционных технологий, но соответствующие издержки неуклонно растут. Поэтому всё большее внимание в процессе промышленного производства программного обеспечения (ПО) фокусируется на технологиях, автоматизирующих все фазы жизненного цикла программного изделия. Несмотря на появление технологий автоматизирующих дизайн и разработку ПО и, как следствие, существенно улучшающих качество ПО, их внедрение идет достаточно сложно и медленно. Это связано не столько с необходимостью дополнительных затрат на внедрение новых технологий, сколько с изменением менталитета программистов, привыкших решать все вопросы проектирования прямо на уровне кода, а не спецификаций. Настоящая работа демонстрирует метод перехода к новым эффективным технологиям разработки ПО.

Современные технологии автоматизации тестирования и разработки кода ПО основаны на процессе генерации кода тестов или приложения. Непременным условием генерации является применения формальных спецификаций в описании системы или ее части, которая должна быть сгенерирована. Использование формальных методов спецификации сразу решает две проблемы:

- проблему наглядного и интуитивно понятного представления требований к системе,
- проблему точного соответствия спецификаций (требований) генерируемому коду системы и тестов.

В рамках формального подхода в качестве языков спецификаций широкое распространение получили языки визуального проектирования UML, SDL, MSC [1-3].

В рассматриваемой технологии для обоснования правильности программ совместно используется верификация и тестирование. Верификация представляет статический метод доказательства правильности символических спецификаций, т.е. описывающих поведение системы графов, дуги которых нагружены символами параметров и атрибутов. Тестирование представляет динамический метод проверки правильности, на основе исполнения множеств конкретных сценариев, полученных из символических путем задания конкретных значений параметрам и атрибутам.

Технологически, процесс верификации реализуется на основании множества спецификаций на языках UML, MSC и SDL. Спецификации, составленные на MSC, используются преимущественно для описания модели окружения и генерации набора тестов. Спецификации на SDL используются для описания динамических и статических свойств системы. Спецификации на UML покрывают описание системы и окружения.

Задачи пилотирования технологии были решены с помощью интегрированного набора из трех мощных инструментов: верификатора, инструмента для автоматической генерации и исполнения тестов и CASE системы Telelogic Tau G2 [4]. К сожалению, тотальное использование технологии генерации кода по спецификациям в современных CASE-системах сдерживается сложностью получения приемлемого по эффективности, реактивности и ряду других характеристик кода. Именно поэтому в рамках описываемой

технологии основное внимание направлено на тестирование, на которое влияние перечисленных ограничений ослабевает.

Применение представленной технологии для верификации и тестирования в области телекоммуникационных систем позволяет добиться следующих результатов:

- повышение качества ПО при использовании интегрированной инструментальной системы верификации и тестирования;
- получение возможности расширения функциональности системы на языках высокого уровня;
- уменьшение затрат на исправление обнаруженных ошибок за счёт их локализации на фазе дизайна в процессе верификации;
- обеспечение полного тестового покрытия поведенческих свойств тестируемой системы с помощью набора автоматически сгенерированных символических трасс, используемых в процессе тестирования;
- автоматическая генерация тестовых наборов и их прогон позволили сократить не менее 50% времени на этапе тестирования.

ЛИТЕРАТУРА:

1. White Paper Using UML 2.0 to Solve Systems Engineering Problems.
2. ITU-T z.100 (08/2002).
3. Recommendation z.120.
4. Telelogic TAU 2.4 UML tutorial.