– недостаточным разделением между кабелями силового электропитания, управления, сигнальными и коммуникационными;

– наличием системы заземления, использующей проводящие каналы, проводники заземления в кабельных желобах и контура заземления.»

Такой подход был предложен более 20 лет назад и в то время являлся компромиссным решением в обеспечении ЭМС. Очевидно, что назрела необходимость использовать современные методы, позволяющие получить количественную оценку уровня электромагнитной совместимости технического средства с окружающей электромагнитной обстановкой (ЭМО). Помехи так же, как и степень помехозащищенности для серии устройств, имеют случайный характер, поэтому логично использовать и вероятностный подход к решению проблемы ЭМС. Количественно оценку уровня ЭМС можно получить, используя выражение для вероятности сбоя устройств

$$P\{x_R < y_N\} = \int_0^\infty f_N(y)\left[\int_0^y f_R(x)dx\right]dy,$$

где $f_N(y)$ и $f_R(x)$ – плотности распределения уровней помех и помехозащищенности устройств.

Для одного конкретного устройства с известной степенью помехозащищенности модель упрощается: вероятность сбоя, которая численно определяет уровень ЭМС, получается из выражения

$$P_{сб} = \int_{U_0}^\infty f_N(U)dU,$$

где U_0 –степень помехозащищенности устройства.

Таким образом можно получить количественную оценку уровня электромагнитной совместимости. Этот подход позволяет, зная вид закона и параметры распределения уровней помех в конкретной ЭМО, установить нормированное значение степени помехозащищенности для обеспечения заданного показателя уровня ЭМС (численное значение $P_{сб}$) еще на стадии проектирования.

В докладе приводятся аналитические зависимости для определения вероятности сбоя систем для композиций наиболее часто встречающихся на практике законов распределения уровней помех и степени помехозащищенности. На основе полученных аналитических зависимостей разработан пакет прикладного программного обеспечения «EMCpar», позволяющий определить как уровень электромагнитной совместимости, так и значения параметров ЭМО, либо степени помехозащищенности при заданном значении $P_{сб}$.

Основной проблемой, сдерживающей распространение такого подхода, является отсутствие регламентированных требований, предъявляемых к уровням помех, и аппаратуры регистрации помех. Имея такую аппаратуру, можно оценить ЭМО, аппроксимировав экспериментальные значения наиболее подходящим законом распределения уровней помех.

Вероятностный подход при решении проблемы ЭМС можно использовать во всех случаях, где возможно статистическое описание характеристик ЭМО и степени помехозащищенности испытуемых устройств. В докладе рассматриваются конкретные примеры использования вероятностных методов решения проблемы ЭМС, полученных в НИЛ «Безопасность и электромагнитная совместимость технических средств » Белорусского государственного университета транспорта.

УДК 681.3

МЕТОДЫ ТЕСТИРОВАНИЯ И ДОКАЗАТЕЛЬСТВА БЕЗОПАСНОСТИ ОБЪЕКТНО-ОРИЕНТИРОВАННЫХ ПРОГРАММНЫХ ПРОДУКТОВ

А. В. ЛОГВИНЕНКО

Белорусский государственный университет транспорта

Внедрение современных микропроцессорных систем в системы управления ответственными технологическими процессами вызвало необходимость доказывать безопасность функционирования не только аппаратных средств, но и программного обеспечения (ПО). На данный момент нет нормативных документов, регламентирующих доказательство объектно-ориентированного ПО на безопасность.

Наиболее эффективным и экономичным способом доказательства безопасности функционирования является тестирование. По существу, тестирование программного обеспечения представляет собой процесс выявления дефектов в программных системах. Дефект может быть привнесен на стадии разработки или сопровождения, в результате чего появляется одна или большее число ошибок. Тестирование, кроме того, подразумевает обнаружение дефектов ПО при некорректной работе аппаратных средств. Тестирование тесно связанно с выявлением и устранением ошибок. Доказательство подразумевает тестирование проектируемого приложения не только на корректность, т.е. соответствие технической документации, но и анализ того, что программное обеспечение реализует заданный техническим заданием алгоритм.

Программное обеспечение – это не только совокупность исходных программных кодов, инструкций и данных, необходимых для выполнения некоторой задачи. Оно включает в себя и модели, построенные в процессе проведения анализа и проектирования. Поэтому программное обеспечение должно подвергаться тестированию во всех его представлениях, т.е. должны проводиться исследования аналитических и проектных моделей программного обеспечения. В этом случае некоторые виды тестирования можно выполнить до начала написания программных кодов.

Объектно-ориентированные возможности в языках программирования вне всяких сомнений оказывают влияние на некоторые аспекты тестирования и доказательства безопасности функционирования. Такие особенности, как наследование классов и интерфейсы поддерживают полиморфизм, по условиям которого программа манипулирует объектами таким образом, что класс, которому они принадлежат, неизвестен. Некоторые языки программирования допускают возможность возникновения одних видов ошибок и устраняют такую возможность для других видов ошибок. Языковые средства, которые поддерживают и вынуждают сокрытие данных, сильно усложняют тестирование.

Например, C++ является строго типизированным, благодаря чему снижается число возможных ошибок в интерфейсах, поскольку компилятор C++ гарантирует соответствие типов формальных и фактических параметров. В Java механизм типов также хорошо развит, но он обладает большей динамикой, нежели C++, в связи с чем его компиляторы менее эффективны при выявлении проблем, обусловленных, например, циклическими ссылками. С другой стороны, в C++ сохраняется возможность наличия в программах ошибок, связанных с применением указателей.

Не только изменения в языке программирования оказывают очевидное воздействие на тестирование, но также изменения процесса разработки программного обеспечения и смещение акцентов при анализе и проектировании. В связи с этим необходим качественный переход от тестирования программных модулей к регрессивному тестированию.

Различия между «старыми» и «новыми» методами разработки вынуждают использовать качественно новые понятия, осуществляется переход от функции к объекту и связям между объектами. Основное различие заключается в том, что объектно-ориентированное программное обеспечение разрабатывается в виде набора объектов, которые по сути дела моделируют задачу. Более того, компоненты, полученные в соответствии с потребностями задачи, могут быть повторно использованы при разработке других программных решений.

Большая польза такого подхода к проектированию заключается в том, что аналитические модели непосредственно отображаются на проектные модели, а те, в свою очередь, отображаются на программы. Следовательно, нужно начинать тестирование на стадии анализа в виде проектного тестирования. Это означает, что процесс тестирования может переплетаться с процессом разработки. Отметим три существенных преимущества применения моделей анализа и проектирования с элементами тестирования:

1 Контрольные примеры (тестовые случаи) могут идентифицироваться на ранних стадиях процесса, даже на стадии определения требований. Раннее тестирование позволяет аналитикам и проектировщикам лучше понимать и четко формулировать требования к системе.

2 Программные ошибки могут быть обнаружены на ранних стадиях процесса разработки, тем самым позволяя эффективнее использовать ресурсы.

3 Тестовые случаи можно проверить на предмет правильности на ранних стадиях проекта.

Некоторые инструментальные средства CASE, например Rational Rose, могут генерировать программные коды на базе проектных моделей. Если сделать предположения, что эти средства работают корректно, можно отметить два основных проявления этого влияния для тестирования объектно-ориентированных программ:

1 Большую часть тестирования требуется выполнять в контексте проектной модели.

2 Применение подхода, ориентированного на тестирование реализаций, требует, чтобы «тестировщик» понимал структуру полученной программы. В этом ему могут оказать помощь инструментальные средства построения профиля программы. Однако если программистам разрешается вносить вручную изменения в генерируемый исходный программный код, то тестирование существенно затрудняется.

Наличие распределенных объектов в современных системах значительно усложняет тестирование и испытания программных средств. Здесь возникает два новых типа ошибок. Параллельные потоки выполнения должны координировать свой доступ к совместно используемым значениям данных. Сбои в синхронизации этих доступов могут привести к тому, что неправильные значения данных будут находиться в памяти даже тогда, когда каждый поток выполняет корректную обработку данных.

Может случиться так, что какой-либо узел распределенной системы перестает правильно функционировать, даже если все другие процессоры работают должным образом. Может выйти из строя то или иное сетевое соединение между узлами, в то время как вся остальная система продолжает исправно выполнять свои функции. Это приводит к отказу системы.

В связи с описанными сложностями проведения испытаний на адекватность объектно-ориентированного программного обеспечения методика тестирования должна быть: скептической – подвергать сомнению качество и требовать его подтверждения; объективной – не отдавать предпочтения никаким предположениям; тщательной – не упускать из виду ни одного важного вопроса; систематической – поиски дефектов должны быть воспроизводимыми.

Методика тестирования программы может быть принята либо тем же программистом, который разрабатывал тестируемый программный продукт, либо другим специалистом, который может обладать независимой точкой зрения как на спецификацию, так и на сам программный продукт.

Для объектно-ориентированного программного обеспечения должны проводиться следующие виды тестирования: тестирование моделей, тестирование классов (вместо тестирования программных модулей), проверка взаимодействия и функционирования компонентов (вместо комплексных испытаний системы), системное тестирование (и испытания подсистем), приемочные испытания, проверка развертывания системы (самотестирование).

Обычно процесс тестирования стоит в списке видов деятельности, выполняемых при разработке программного обеспечения, последним, сразу же за *реализацией*. Этот вид деятельности относится к тем типам тестирования, которые проверяют, соответствует ли программный продукт как единое целое по своим функциональным возможностям тому, чем он должен являться. Предлагается проводить тестирование в различные моменты разработки, а не только на завершающей стадии. Испытаниям должен быть подвергнут не только программный код, но и должны проводиться исследования аналитических и проектных моделей программного обеспечения.

УДК 621.396.67.061

СИНТЕЗ РАДИОГОЛОГРАФИЧЕСКИХ АНТЕНН

В. Н. МИЗГАЙЛОВ, Д. Н. ЗЕЛИНСКИЙ
Белорусский государственный университет транспорта

Рассматриваются задачи построения относительно малоизвестного класса антенн-радиоголографических. Название антенн следует из их схемы построения: облучатель и радиоголограмма.

Голограмма – это интерференционная система, построенная путем полной записи (с учетом амплитуд, фаз, поляризации и т.п.) взаимодействующих пучков оптических сигналов. Радиоголограммой называется аналогичная система, но построенная в диапазоне радиоволн.

Голограмма всегда рассматривалась как оптическая система, способная воспроизводить записанные на ней сигналы, если её облучать одним из участвующих в записи пучков волн. При воспроизведении записанных на ней сигналов она обеспечивает сохранение направления движения всех пучков волн. Если голограмму облучать сопряженными сигналами, то направления движения пучков восстанавливаемых волн меняется на обратное тем, которые были при записи. Осуществив