УДК 004.052.42

В.В. СЕРГИЕНКО[1], Б.М. КОНОРЕВ[2], Л. НОВЫ[3], Г.Н. ЧЕРТКОВ[1]

[1]*Сертификационный центр АСУ, Харьков, Украина;*
[2]*Национальный аэрокосмический университет им. Н.Е. Жуковского "ХАИ", Украина*
[3] *ZAT, Czech Republic*

КАЛИБРОВКА МЕТОДОВ ИЗМЕРЕНИЯ ИНВАРИАНТОВ КРИТИЧЕСКОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ: ПРОФИЛЬ ИНЪЕКТИРУЕМЫХ ТЕСТОВЫХ ДЕФЕКТОВ

Информативность испытаний и подтверждение приемлемых уровней вероятности скрытых дефектов при независимой верификации критического программного обеспечения обеспечивается с помощью адекватного профиля дефектов и калибровок методов, используемых при испытаниях. Рассматривается разработка модели профиля дефектов для калибровки методов измерения инвариантов при статическом анализе.

программное обеспечение, оценивание, независимая верификация, статический анализ, инвариант, калибровка, скрытый дефект, профиль дефектов

Введение

Базовые характеристики качества систем Гарантоспособность (надежность, готовность, обслуживаемость) и Безопасность в различных сферах критической инженерной деятельности существенно зависят от характеристик программного обеспечения (ПО), реализующего критические функции систем. Скрытые дефекты критического ПО являются факторами риска аномального функционирования или отказов систем. В силу этого разработка ПО систем критического применения представляет важный объект нормативного регулирования. В качестве обязательного требования нормативной базой (общепромышленными и отраслевыми стандартами) таких систем предусматривается проведение квалификационных испытаний с независимой верификацией и валидацией критического ПО. В этом контексте независимость означает эффективное использование принципа технологического и административного разнообразия и служит для повышения уровня бездефектности и снижения рисков аномального (аварийного) поведения системы в целом из-за скрытых дефектов критического ПО. **Прогноз вероятности скрытых дефектов** является одним из главных результатов независимой верификации.

Степень неопределенности (достоверность) испытаний существенно зависит от полноты и эффективности реализации принципа разнообразия. Это обуславливает высокую актуальность проблемы доказательности результатов независимой верификации и получения достоверных количественных оценок характеристик качества критического ПО.

Сценарий целевой технологии

Общий подход для организации и проведения испытаний при независимой верификации реализуется с помощью целевой технологии [1].

Сценарий целевой технологии независимой верификации представляет сеть взаимодействующих процессов, реализующих три базовых методики:

– нормализация проекта ПО как объекта экспертизы [2];

– измерение инвариантов (неизменных свойств ПО) и оценка характеристик качества ПО [3, 4];

– калибровка; интегральная оценка; рентабельность [5].

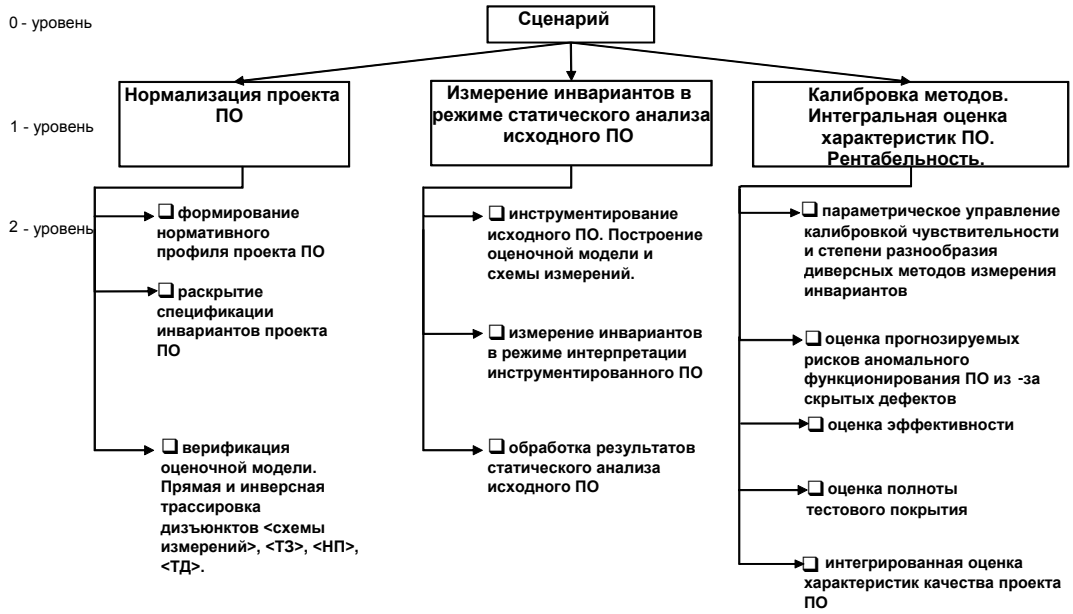

Рис. 1. Дерево узлов функциональной IDEF0-модели сценария целевой технологии

Функциональная модель сценария (см. рис.1) разрабатывается на основе методологии IDEF0 моделирования и представляет иерархию моделей различных уровней детализации процессов сценария. Проведение сценария поддерживается на аналитическом, информационном и организационном уровнях.

Методика «Нормализация проекта ПО как объекта экспертизы» включает рабочие пакеты:

– формирование и верификацию нормативного профиля (опорной модели) для оцениваемого проекта;

– раскрытие спецификаций измеряемых атрибутов проекта ПО и формирование схемы измерений;

– прямую и инверсную трассировку элементов проекта ПО и схемы измерения инвариантов;

– оценку полноты тестового покрытия проекта ПО.

Методика «Измерение инвариантов и оценка характеристик качества ПО» включает: инструментирование исходного ПО – формирование оценочной модели пригодной для измерения инвариантов;

– измерение семантических, интервально – точностных и логических инвариантов в режиме интерпретации инструментированной версии ПО;

– обработку результатов измерения инвариантов и диагностирование дефектов ПО.

Методика «Калибровка. Интегральная оценка Рентабельность» включает:

– калибровку чувствительности и степени разнообразия методов измерения инвариантов ПО;

– оценку степени неопределенности измерения инвариантов и вероятности скрытых дефектов ПО;

– интегральную оценку характеристик гарантоспособности и функциональной безопасности.

Целевая технология позволяет сделать процесс верификации доказательным и независимым. Доказательность достигается за счет использования строгих методик с калибровкой полученных результатов и получением количественных значений испытуемых характеристик. Независимость обеспечивается не только организационными и административными мероприятиями, но и путем использования диверсных (отличных от тестов разработчика) методов испытаний. Технология позволяет спрогнозировать вероятность остаточных (скрытых) дефектов критического ПО и до-

биться желаемых значений показателей как гарантоспособности так и безопасности.

Технология основана на использовании усовершенствованной методологии статического анализа исходных текстов ПО, и обеспечивает решение следующих задач:

– повышение достоверности оценок характеристик ПО за счет доказательной (измеряемой) реализации принципов технологического разнообразия (диверсности) на основе диверсифицированного измерения семантических, интервально-точностных, логических и др. инвариантов ПО (физических или абстрактных свойств ПО, остающихся неизменными по определению в течение жизненного цикла) на платформе статического анализа исходного ПО;

– прогнозирование вероятности скрытых дефектов ПО на основе экспериментальной калибровки чувствительности и степени разнообразия диверсных методов измерения инвариантов в условиях конкретного проекта ПО, методом посева тестовых дефектов;

– управление эффективностью (путем минимизации расходуемых ресурсов) прогнозирования вероятности скрытых дефектов с использованием индикатора снижения вероятности остаточных дефектов ПО в процессе реализации композиции диверсных методов измерения инвариантов.

– оценку полноты тестового покрытия критического ПО методом прямой и обратной трассировки для оценки отношений эквивалентности и импликативности множеств дизъюнктов: а) опорной (ссылочной) и оценочной моделей качества ПО; б) спецификации технических требований к ПО); в) проектных определений и обоснований (проектно-конструкторской документации ПО).

Модель профиля дефектов

Необходимым элементом сценария целевой технологии является тестовый профиль дефектов, с помощью которого производится калибровка методов измерения инвариантов посредством посева тестовых дефектов.

Профиль дефектов (ПД) – это классификационная схема дефектов (определяет типы дефектов), с учетом вероятностей появления каждого типа дефектов.

Модель ПД должна позволять адекватно классифицировать дефекты в соответствии с реальными программами для данного класса задач.

Для построения модели ПД, учитывающей все множество дефектов, необходимо учесть следующие характеристики:

– аппаратно-программная реализация;

– функциональные характеристики;

– стадии разработки в течении жизненного цикла ПО;

– адресное поле (исполнимый код);

– исходный код.

ПД должен учитывать:

– уровень внесения дефекта (соответственно, этот же уровень является уровнем исправления). Поскольку дефект вносится программистом, то данный уровень определяется в терминах исходного кода;

– уровень проявления дефекта. Дефект проявляется и может быть обнаружен либо на этапе тестирования (испытаний) – информацию дает исполнимый или исходный код, либо непосредственно при работе программы – исполнимый код;

– уровень определения чувствительности к дефекту используемых методов (калибровка через посев). Необходим для калибровки используемых методов испытаний и как результат для получения частной и обобщенной оценок характеристик экспертируемого ПО.

Рассмотрение имеющихся на данный момент классификационных схем дефектов на различных стадиях жизненного цикла ПО:

– схема Канера [6];

– схема «тестировщиков» [7];

– схема Эндреса [8];
– схема Базили/Перрикоун [9];
– схема Остранда/Вейкера [10];
– схема Гудэнафа/Джерхарта [11];
– схема Кнута [12] и т.д.

позволяет выявить следующие существенные недостатки, присущие этим (и многим другим) классификациям: они неоднозначны, избыточны и сложны для автоматизации. Для каждой схемы, можно найти ошибки, которые могут быть классифицированы в более чем одну категорию.

Кроме того, каждая из этих схем требует наличия человека для ручной классификации ошибок. В этой ситуации могут быть внесены свои ошибки при классификации, плюс требуются дополнительные трудозатраты.

При построении ПД целесообразно выбрать классификационную схему, предложенную Демилло и Мэтью [13], которая лишена приведенных выше недостатков.

Схема Демилло и Мэтью (далее – схема Д/М) строится в терминах исходного кода.

Для математической строгости и однозначности дефектом считается подстрока в коде, которую нужно изменить (удалить, исправить, перенести в другое место, добавить) в ошибочной программе для получения корректной программы.

Математически схема представляется следующим образом:

Программа представляется последовательностью строк $x_1y_1x_2y_2....x_ky_k$, причем любое y_i может быть пустой строкой. Если для получения корректной программы необходимо заменить y_i на y_i`; $1 < i < k$, то дефектом считается строка $y_i$ если она не пустая или строка $y_i$` если y_i пустая.

Категории дефектов моделируются с помощью синтаксических преобразователей. Синтаксический преобразователь в терминах грамматики исходного кода показывает изменение каждой подстроки некорректной программы в корректную. Т.е. для корректной строки он сохраняет ее значение, а для некорректной (содержащей дефект) указывает какое изменение нужно произвести, чтобы исправить дефект.

Схема Д/М содержит 4 главных категории:

1. Ошибочно расположенный объект (преобразователь T^p)

К этой категории относятся дефекты, исправление которых требует изменения расположения подстроки в коде.

2. Пропущенный (Отсутствующий) объект (преобразователь T^m)

К этой категории относятся дефекты, исправление которых требует вставки синтаксического объекта в неправильную программу. Пропущенным объектом может быть последовательность инструкций, единственная инструкция, выражение, или единичный оператор. Эти четыре синтаксических объекта формируют подкатегории «отсутствующего объекта».

3. Избыточный объект (преобразователь T^s)

К этой категории относятся дефекты, исправление которых требует удаления ошибочной подстроки.

4. Некорректный объект (преобразователь T^i)

К этой категории относятся дефекты, не попавшие в предыдущие категории.

Для возможности автоматизации и отсутствия избыточности при классификации категории ранжированы по приоритету (в порядке убывания): T^p, T^m, T^s, T^i. Если дефект попал в категорию с более высоким приоритетом, то в категорию с более низким приоритетом он уже не попадает.

Механизм учета дефектов для диалекта языка Pascal и принципы его построения являются открытыми. Математически доказана полнота покрытия и возможность построения автоматизированного учета и классификации ошибок и для других языков программирования.

Использование классификационной схемы, основанной на грамматике исходного кода:

– позволяет реализовывать проверки в терми-

нах исходного кода, таким образом, сразу можно определять область покрытия кода проверками;

– воссоздается процедура внесения ошибок, – именно в исходном коде программист вносит дефекты, которые приводят к ошибкам;

– легко реализуема процедура засева дефектов для калибровок;

– возможна автоматизация сбора ошибок/дефектов.

В силу вышеизложенного классификационная схема Д/М может использоваться для получения полного и неизбыточного тестового ПД.

Полнота тестового профиля дефектов

Для определения полноты и неизбыточности тестового ПД устанавливается все множество возможных типов дефектов (D_{all}) исходного кода для конкретного проекта ПО:

$$D_{all} = \bigcup_i (G_i x M_i),$$

где G_i – i-й тип грамматической конструкции исходного кода ПО. Полный набор типов конструкций («спектр операций») определяется на основании предварительного анализа конкретного проекта ПО.

M_i – множество возможных искажений для i-й конструкции. Конкретный набор M_i определяется на основе возможных искажений допускаемых (не определяемых как ошибка времени компиляции) компилятором для данной конструкции.

Тестовый ПД устанавливается исходя из проверочных способностей моделей измерения инвариантов. Для проверки каждого инварианта строится отдельная модель ПО, которая содержит только те характеристики всего проекта, которые необходимы для оценки требуемого инварианта конкретным методом. Измерение инвариантов производится в терминах построенных моделей ПО. Для каждой модели определяется подмножество типов дефектов D_{mod}, возможных для данной модели. Множество типов дефектов для моделей, использованных при проверке инвариантов, вычисляется как:

$$D_{test} = \bigcup D_{mod},$$

и определяет все множество типов дефектов тестового ПД.

Полнота и избыточность тестового профиля дефектов определяется путем анализа множеств D_{all} и D_{test}. Возможные случаи:

1) $D_{test} \setminus D_{all} \cap D_{test} \neq \varnothing$,

тестовый ПД избыточный (содержит типы дефектов, не свойственные проекту) и необходима его коррекция для исключения ненужных проверок.

2) $D_{all} = D_{test,,}$

идеальный вариант. Тестовый ПД является полным и неизбыточным.

3) $D_{test} \subset D_{all}$, $D_{all} \not\subset D_{test}$.

наиболее типичный случай. Построенный тестовый ПД неполный.

Степень полноты тестового ПД (F) в общем случае определяется как отношение мощностей множеств D_{test} и D_{all}:

$$F = \frac{|D_{all} \setminus D_{test} \cap D_{all}|}{|D_{all}|}.$$

Использование профиля дефектов при калибровке

Определение полноты покрытия, чувствительности и степени разнообразия методов измерения инвариантов для конкретного проекта ПО производится с помощью калибровки, предусматривающей искусственный посев (внесение, инъекцию) тестовых дефектов.

Калибровка производится с учетом специфики конкретного проекта ПО, представленной спектром (процентным составом) компьютерных операций, измеренным для конкретного проекта при статическом анализе исходных текстов ПО. Используется модифицированный метод «посева». Его суть состоит в «капельной инъекции» одиночного тестового дефекта определенного типа и циклического вы-

полнения процедуры «инъекция – обнаружение дефекта» статистически значимое число раз.

Результатом калибровки является определение чувствительности *j*–го метода к *i*-му типу дефектов из подмножества дефектов – парциальная чувствительность D_{ij} (множество остаточных дефектов). Дефекты, посеянные и необнаруженные при калибровке называются остаточными.

Интегральная (полная) чувствительность метода ко всем типам дефектов из засеваемого подмножества является объединением парциальных чувствительностей:

$$D_j = \bigcup_i D_{ij} .$$

Калибровка позволяет сравнить методы попарно и определить максимально эффективные наборы проверок с помощью вычисления степени разнообразия методов для конкретного проекта.

Оценка степени разнообразия 2-х использованных методов измерения инвариантов (диверсность), множества остаточных дефектов которых определяются как D_1 и D_2, производится по формуле:

$$P = \frac{|D_1 \bigcap D_2|}{|D_1 \bigcup D_2|}.$$

При этом нужно учитывать, что если D_1 является подмножеством D_2 ($D_1 \subset D_2$), то использование 2-го метода после первого не имеет смысла.

Общая чувствительность всех использованных методов:

$$D_o = \bigcap_j D_j$$

Общая чувствительность композиции использованных методов измерения инвариантов является конечной целью экспериментальной калибровки.

Определение D_o позволяет уточнить область возможного нахождения множества скрытых дефектов $D_{скр}$: поскольку все скрытые дефекты, находящиеся вне множества D_o определяются и устраняются при проверке инвариантов, можно утверждать что $D_{ñêð} \subset D_o$.

Заключение

Проведен аналитический анализ существующих классификационных схем дефектов на различных стадиях жизненного цикла ПО. Предложена модель тестового профиля дефектов.

Предложенная модель профиля дефектов позволяет выполнить калибровку и интегральную оценку методов измерения инвариантов. Использование тестового профиля дефектов при калибровке методов измерения инвариантов критического программного обеспечения при независимой верификации дает следующие преимущества:

1) повышается достоверность испытаний – адекватно проверяются и оцениваются требуемые характеристики программного продукта;

2) количественно прогнозируется вероятность остаточных (скрытых) дефектов;

3) повышается качество и эффективность экспертизы путем использования оптимального набора методов (через учет взаимной чувствительности и полноты методов);

4) определяется полнота проверок.

Литература

1. Конорев Б.М., Алексеев Ю.Г., Сергиенко В.В., Харченко В.С., Чертков Г.Н. Целевая технология рентабельной оценки надежности и функциональной безопасности критического программного обеспечения // Радіоелектронні і комп'ютерні системи. – 2007. – № 9(27). – С. 162-170.

2. Конорев Б.М., Сергиенко В.В., Чертков Г.Н. Оценивание качества ПО ИУС критического применения: нормализованное представление объекта экспертизы // Системы контроля и управления технологическими процессами: Сборник научных статей. – Луганск: Світлиця, 2006. – С. 385-390.

3. Конорев Б. М., Алексеев Ю.Г., Засуха С.А., Манжос Ю.С., Семенов Л.П., Сергиенко В.В., Хар-

ченко В.С., Чертков Г.Н. Модель оценивания качества ПО ИУС критического применения на основе инвариантов // Радиоэлектронные и компьютерные системы. – 2006. – № 7. – С. 162-170.

4. Конорев Б.М., Манжос Ю.С. и др. Оценивание качества программного обеспечения информационно-управляющих систем критического применения: утилиты семантического и интервально-точностного анализа исходного кода // Труды Межд. Симпозиума “Измерения важные для безопасности в реакторах”. – Словакия, Смоленице. – 2005. – С. 8-18.

5. Конорев Б.М., Сергиенко В.В., Манжос Ю.С. и др. Калибровка чувствительности методов статического анализа, используемых для оценки качества и безопасности ПО ИУС АЭС // Труды Международного Симпозиума “Измерения важные для безопасности в реакторах”. – Москва, 23-25 ноября 2004. – С. 12-15.

6. Канер С., Фолк Д., Нгуен К.. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений. – М.: «ДиаСофт», 2001. – 360 с.

7. Открытая схема тестировщиков. [Электронный ресурс]. – Режим доступа: http://void.ru/?do=printable&id=702.

8. Endres. A. An Analysis of Errors and Their Causes in System Programs // IEEE Transactions on Software Engineering. – Vol. SE-1, No. 2. – P. 140-149.

9. Basili V.R., Perricone B.T.. Software Errors and Complexity: An Empirical Investigation // Comm. of the ACM. – Vol. 27, No. 1. – P. 42-52.

10. Ostrand T.J., Weyuker E.J. Collecting and Categorizing Software Error Data in an Industrial Environment // The Journal of Systems and Software. – Vol. 4. – P. 289-300.

11. Goodenough J.B., Gerhart S.L. Toward a Theory of Test Data Selection // IEEE Transactions on Software Engineering. – Vol. SE-1, No. 2. – P. 156-173.

12. Knuth D.E. The Errors of T E X // Software Practice and Experience. – Vol. 19, No.7. – P. 607-685.

13. DeMillo R., Mathur A. A Grammar Based Fault Classification Scheme and its Application to the Classification of the Errors of TEX. – Software Engineering Research Center and Department of Computer Sciences, Purdue University, W. Lafayette. – IN 47907, 1995.

Поступила в редакцию 16.02.2008

Рецензент: д-р техн. наук, проф. В.М. Илюшко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.