ИНФОРМАТИКА

УДК 004.05

МЕТОДИКА ОЦЕНКИ НАДЕЖНОСТИ ПРОГРАММНЫХ СРЕДСТВ

А.В. АНЦЫПОВ, В.В. БАХТИЗИН

Белорусский государственный университет информатики и радиоэлектроники
П. Бровки, 6, Минск, 220013, Беларусь

Поступила в редакцию 8 декабря 2009

Предлагается методика оценки надежности программных средств, основанная на: оценке надежности отдельно взятого модуля, оценке веса модуля, определении метрик для оценки веса модуля и отображении изменения надежности модулей в графическом виде.

Ключевые слова: качество программных средств, надежность программных средств, оценка надежности программных средств.

Введение

Сегодня деятельность многих организаций и предприятий напрямую зависит от правильной обработки информации соответствующими программными средствами. Поэтому к качеству современных программных средств предъявляются высокие требования.

Одним из основных стандартов качества в области инженерии программного обеспечения в настоящее время является стандарт ISO/IEC 9126-1:2001 — Software engineering — Product quality (Информационная технология. Оценка программного продукта) [1].

Данный стандарт регламентирует шесть основных характеристик качества программных средств: функциональность, надежность, эффективность, практичность, сопровождаемость, мобильность.

Оценка надежности модуля

Надежность является одной из основных характеристик качества программных средств [1, 3]. Оценка надежности разрабатываемого программного средства является одной из важных задач, стоящих перед разработчиком программного средства. Оценка надежности всего программного средства является достаточно сложной задачей. Существенно проще оценить надежность отдельно взятых модулей, а потом дать общую оценку надежности всего программного средства.

При оценке надежности модуля необходимо учитывать не только количество ошибок, но также и последствия, которые могут произойти в результате возникновения ошибки, и вероятность ввода *k*-го набора данных, приведшего к ошибке. В соответствии с международным стандартом IEEE 1044.1-1995 Guide to Classification for Software Anomalies (Классификация ошибок программного средства) [2] все ошибки можно разбить на шесть основных групп: Priceless — ошибка, после возникновения которой модуль перестает функционировать; High — ошибка, после возникновения которой модуль функционирует, но функционирует неверно; Medium — ошибка, после возникновения которой пользователь может продолжать использовать модуль, но данная ошибка может нанести существенный урон данным; Low — ошибка, последствия которой носят незначительный характер; None — ситуация, когда одна часть пользователей считает, что это нормальное поведение модуля, а вторая часть, что произошла ошиб-

ка; Detrimental — ситуация, когда возможно появление ошибки (примером такой ситуации может быть постоянно растущий объем оперативной памяти, используемой модулем). Кроме того, следует учесть ситуацию, когда ошибка при вводе *k*-го набора данных может не возникнуть.

В статье предлагается следующая методика определения оценки надежности модуля. Каждому типу ошибки экспертным путем присваивается вес E_j, условие нормирования для E_j имеет вид $\sum_{j=0}^{6} E_j = 1$. E_0 соответствует ситуации отсутствия ошибки. Очевидно, что значение E_0 в условии нормирования равно нулю. Чем выше вес ошибки, тем более серьезные последствия она несет. Тогда оценка надежности *i*-го модуля p_i определяется по формуле

$$p_i = 1 - \sum_{k=1}^{D} (E_j \cdot b_k), \quad (1)$$

где D — количество наборов данных, введенных пользователем; E_j — вес ошибки, произошедшей после ввода *k*-го набора данных; причем $j \in \{0, 1, 2, 3, 4, 5, 6\}$; b_k — вероятность ввода *k*-го набора данных.

Условие нормирования для b_k имеет вид $\sum_{k=1}^{D} b_k = 1$.

Приведем пример. Банковскую компьютерную систему можно рассматривать как программное средство, где в качестве модулей выступают банкоматы, инфокиоски, серверы, обрабатывающие запросы пользователей, и прочее. В качестве исследуемого модуля возьмем банкомат. Пользователь банкомата может выполнить три операции: просмотр баланса на счету, снятие денег со счета и оплата коммунальных услуг. Пусть вероятность просмотра баланса равна 0,1, вероятность снятия денег равна 0,65 и вероятность оплаты коммунальных услуг равна 0,25. Допустим, что просмотр баланса произошел без ошибок, снятие денег произошло с ошибкой типа High (банкомат снял деньги со счета, но не выдал их пользователю), а оплата коммунальных услуг произошла с ошибкой типа Low (коммунальные услуги были оплачены, но пользователю не был выдан чек). Вес ошибок Priceless, High, Medium, Low, None, Detrimental соответственно равны 0,4; 0,25; 0,2; 0,1; 0,04; 0,01. Тогда оценка надежности модуля будет равна

$$p_i = 1 - (0 \cdot 0{,}1 + 0{,}25 \cdot 0{,}65 + 0{,}1 \cdot 0{,}25) = 0{,}8125.$$

При оценке надежности всего программного средства необходимо учитывать, что различные модули в различной степени влияют на надежность программного средства (например, ошибка, возникшая при сохранении информации более критична, чем ошибка, возникающая при ее отображении).

Введем понятие веса модуля как степени влияния отказа модуля на отказ всего программного средства.

Обозначим вес *i*-го модуля как s_i. Условие нормирования для s_i имеет вид $\sum_{i=1}^{n} s_i = 1$, где n — количество модулей в программном средстве. Чем ближе значение s_i к единице, тем больше отказ *i*-го модуля влияет на отказ всего программного средства.

Оценка надежности разрабатываемого программного средства $P_{\text{ПС}}$ определяется по формуле

$$P_{\text{ПС}} = \sum_{i=1}^{n} (s_i \cdot p_i). \quad (2)$$

Метрики оценки веса модуля

Предлагается следующий набор метрик для оценки веса модуля. При этом в метриках, основанных на сравнении с базовыми аналогами (метрики 5, 6, 7), будем считать, что аналоги

разработаны персоналом такой же квалификации и при той же организации процесса разработки.

Пусть программа содержит несколько модулей.

1) Метрика "Потребляемые ресурсы модулем". Как правило, более весомый модуль потребляет и больше ресурсов.

Если обозначить: A — объем потребляемых ресурсов модулем (объем оперативной памяти и т.п.); B — общий объем потребляемых программным средством ресурсов, то числовую оценку метрики можно найти по формуле $X=A/B$. При этом справедливо: $0<A<B$; $0<X<1$.

2) Метрика "Время работы модуля". Как правило, чем большее время находится модуль в работе, тем он более весом.

Если обозначить: A — время работы модуля; B — общее время выполнения всей задачи, то числовую оценку метрики можно найти по формуле $X=A/B$. При этом справедливо $0<A<B$; $0<X<1$.

3) Метрика "Частота использования модуля". Вероятность появления ошибки возрастает с ростом количества вызовов модуля, поэтому часто вызываемые модули более весомы, нежели модули, вызываемые реже.

Если обозначить: A — количество вызовов модуля; B — общее количество всех вызовов модулей, то числовую оценку метрики можно найти по формуле $X=A/B$. При этом справедливо $1\le A<B$; $1/B<X<1$.

4) Метрика "Зависимость модулей". Как правило, в программном средстве модули связаны в древовидную структуру, где неправильная работа одного модуля приводит к неправильной работе других модулей.

Если обозначить: A — количество модулей, зависящих от данного модуля; B — общее количество всех модулей, то числовую оценку метрики можно найти по формуле $X=A/B$. При этом справедливо $0\le A<B$; $0<X<1$.

5) Метрика "Объем модуля". Как правило, наиболее весомые модули имеют больший объем кода.

Если обозначить: A — объем модуля (например, число строк кода); B — объем кода аналогичного по функциональности модуля, взятый из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A >= B. \end{cases}$$

При этом справедливо $0<A$; $0<X\le 1$.

6) Метрика "Команда разработки модуля". Как правило, в разработке наиболее весомых модулей принимает участие большее количество разработчиков.

Если обозначить: A — количество разработчиков, принимающих участие в разработке модуля; B — количество разработчиков, принимавших участие в создании аналогичного по функциональности модуля, взятое из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A >= B. \end{cases}$$

При этом справедливо $0<A$; $0<X\le 1$.

7) Метрика "Время, затраченное на разработку модуля". Как правило, на разработку наиболее весомого модуля затрачивается больше времени.

Если обозначить: A — время, затраченное на разработку модуля; B — время затраченное на разработку аналогичного по функциональности модуля, взятое из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A >= B. \end{cases}$$

При этом справедливо $0<A$; $0<X\leq 1$.

8) Метрика "Квалификация команды разработчиков". Как правило, в разработке наиболее весомых модулей принимают участие более квалифицированные разработчики.

Если обозначить: A — уровень квалификации команды разработчиков, принимающих участие в разработке модуля (A — может быть определенно экспертным путем, $0<A\leq 1$); B — уровень квалификации команды разработчиков, принимавших участие в создании аналогичного по функциональности модуля, взятый из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A \geq B. \end{cases}$$

При этом справедливо $0<A\leq 1$; $0<X\leq 1$.

9) Метрика "Документирование модуля". Как правило, более весомые модули должны иметь больший объем документации.

Если обозначить: A — суммарный объем документов, описывающих анализируемый модуль; B — суммарный объем документов аналогичного по функциональности модуля, взятый из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A \geq B. \end{cases}$$

При этом справедливо $0<A$; $0<X\leq 1$.

10) Метрика "Тестовые наборы модуля". Как правило, более весомые модули имеют большее количество тестовых наборов данных.

Если обозначить: A — количество тестовых наборов разрабатываемого модуля; B — количество тестовых наборов аналогичного по функциональности модуля, взятое из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A \geq B. \end{cases}$$

При этом справедливо $0<A$; $0<X\leq 1$.

11) Метрика "Время тестирования модуля". Как правило, более весомые модули тестируются более продолжительное время.

Если обозначить A — время тестирования разрабатываемого модуля; B — время тестирования аналогичного по функциональности модуля, взятое из статистических данных организации-разработчика программного средства, то числовую оценку метрики можно найти по формуле

$$X = \begin{cases} \dfrac{A}{B}, & \text{если } A < B, \\ 1, & \text{если } A \geq B. \end{cases}$$

При этом справедливо $0<A$; $0<X\leq 1$.

Оценка веса модуля

Вес i-го модуля s_i определяется по формуле

$$s_i = \frac{\sum_{j=1}^{m_i} (V_{ij} \cdot X_{ij})}{\sum_{j=1}^{m_i} V_{ij}}, \qquad (3)$$

где $\sum_{j=1}^{m} V_{ij} = 1$; m — общее количество метрик; m_i — количество метрик, используемых для расчета значения веса i-го модуля ($m_i \le m$); X_{ij} — значение j-й метрики, относящейся к i-му модулю; V_{ij} — весовой коэффициент j-й метрики, относящейся к i-му модулю, определяющийся по статистическим данным организации разработчика программного средства или экспертным путем.

Зная надежность модулей p_i и их веса s_i, по формуле (2) можно найти надежность всего программного средства.

Отображение изменения надежности модулей в графическом виде

Разрабатываемые модули можно описать распределением модулей по уровням надежности и представить в виде матрицы-столбца (вектора надежности) $\mathbf{F}=[f_i]_{r\times 1}$, где r — число уровней надежности модуля. Численное значение элементов вектора надежности $\mathbf{F}$ показывает процент модулей, относящихся к i-му уровню надежности. К i-му уровню надежности будут принадлежать модули с надежностью в диапазоне $[(i–1)/r; i/r)$. При этом f_1 — процент модулей, уровень надежности которых близок к нулю; f_r — процент модулей с уровнем надежности, близким к 1.

Условие нормирования вектора надежности имеет вид $\sum_{i=1}^{r} f_i = 1$.

В процессе итеративной разработки, как правило, количество ошибок модуля уменьшается, что приводит к росту надежности модулей и изменению элементов вектора надежности.

Пусть $\mathbf{F}_1$, $\mathbf{F}_2$ — соответственно вектора надежности до и после очередной итерации разработки; $B=[b_{ij}]_{r\times r}$ — матрица изменения надежности модулей. Значение элемента матрицы $[b_{ij}]_{r\times r}$ показывает процент модулей, которые, находясь на j-м уровне надежности, после очередной итерации разработки перешли на i-й уровень.

Вектор $\mathbf{F}_2$ будет равен матричному произведению матрицы B и вектора $\mathbf{F}_1$.

$$\mathbf{F}_2 = B \times \mathbf{F}_1.$$

Пусть разрабатываемое программное средство состоит из 100 модулей, число уровней надежности равно пяти и вектор $\mathbf{F}_1$ равен $[0{,}2;\ 0{,}15;\ 0{,}25;\ 0{,}15;\ 0{,}25]_{5\times 1}$. Пусть на следующей итерации 5 модулей, находившихся на первом уровне, остались на нем же, 5 модулей перешли с первого уровня на третий, а 10 модулей перешли с первого уровня на четвертый. Тогда первый столбец матрицы B можно представить в виде

$$\left[\frac{5}{0{,}2\cdot 100}; 0; \frac{5}{0{,}2\cdot 100}; \frac{10}{0{,}2\cdot 100}; 0\right] = 0{,}25; 0; 0{,}25; 0{,}5; 0\ .$$

Аналогичным образом можно сформировать оставшиеся столбцы (в процессе разработки реального программного средства процесс генерации матрицы B можно автоматизировать, что избавит разработчика от монотонной работы).

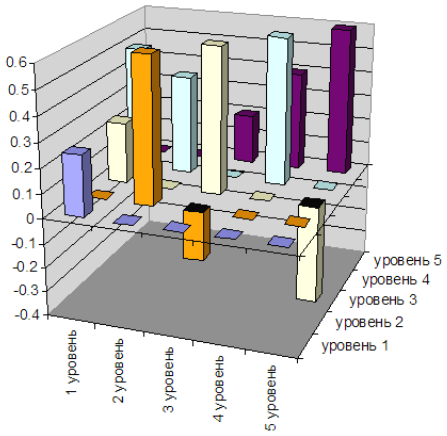
Матрица B для данной итерации равна

$$\begin{bmatrix} 0,25 & 0 & 0 & 0 & 0 \\ 0 & 0,6 & 0,2 & 0 & 0 \\ 0,25 & 0 & 0,6 & 0 & 0,4 \\ 0,5 & 0,4 & 0 & 0,6 & 0 \\ 0 & 0 & 0,2 & 0,4 & 0,6 \end{bmatrix}_{5\times5}$$

Тогда вектор $\mathbf{F}_2$ будет равен $[0{,}05;\ 0{,}14;\ 0{,}3;\ 0{,}25;\ 0{,}26]_{5\times1}$.

В графическом виде матрица *B* представлена на рисунке, из которого видно, что у 60% модулей, находящихся на третьем уровне надежности, после очередной итерации надежность не изменилась, у 20% модулей надежность улучшилось, и они перешли с третьего уровня на пятый, и у 20% модулей надежность ухудшилась, и они перешли с третьего уровня на второй.

Матрица *B* показывает изменение надежности программного средства в динамике. И дает более детальное представление об изменении надежности разрабатываемого программного средства.



Графическое представление матрицы изменения надежности

Заключение

В данной работе предлагается методика оценки надежности программного средства, основанная на оценке надежности отдельно взятого модуля, оценке веса модуля, определении метрик для оценки веса модуля, а также представление изменений надежности модулей в графическом виде.

Используя предложенную методику, у разработчика появляется возможность оценивать и контролировать надежность разрабатываемого программного средства. Достоинствами предложенной методики является то, что она основана на общепризнанных мировых стандартах и проста в использовании.

METHOD OF ESTIMATION OF SOFTWARE RELIABILITY

A.V. ANTSYPOV, V.V. BAKHTIZIN

Abstract

This paper suggests a method of reliability estimation of software. It is based on: single module reliability estimation; module weight estimation, metrics definition for module weight estimation, and method of graphical presentation of module reliability variation.

Литература

1. ISO/IEC 9126-1:2001 Software engineering — Product quality.
2. IEEE Std 1044.1-1995 Guide to Classification for Software Anomalies.
3. *Бахтизин В.В., Глухова Л.А.* Стандартизация и сертификация программного обеспечения. Минск, 2006.