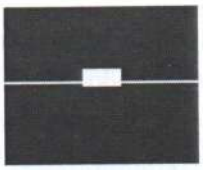

ПОРТФЕЛЬ РЕДАКЦИИ

БИТ

I.M. Azymshin, V.O. Chukanov

Definition Of Software Reliability, Taking Into Account The Possibility Of New Errors WhenRepairing Previously Detected

Keywords: software reliability

In today's world to determine the reliability of software are widely used Jelinski-Moranda and Schick-Wolverton reliability models. But these models do not take into account the possibility that the developer can allow the appearance of new errors in correction of previously detected errors. The article discusses the option of modification of these models, taking into account the probability of emergence of error in correcting previously detected, that did not be before finalize the software. The article also contains studies of the probability of new errors when removing previously detected.

И.М. Азымшин, В.О. Чуканов

ОПРЕДЕЛЕНИЕ НАДЁЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С УЧЁТОМ ВОЗМОЖНОСТИ ПОЯВЛЕНИЯ НОВЫХ ОШИБОК ПРИ ИСПРАВЛЕНИИ РАНЕЕ ОБНАРУЖЕННЫХ

Рассмотрим модель Джелинского–Моранды. Это модель с дискретным убыванием интенсивности отказов, где предполагается, что интенсивность ошибок описывается кусочно-постоянной функцией, пропорциональной числу неустранённых ошибок. Т.е. предполагается, что интенсивность отказов λ(t) постоянна до обнаружения и исправления ошибки, после чего она опять становится постоянной, но с другим, меньшим, значением. При этом предполагается, что между λ(t) и числом оставшихся в программе ошибок существует прямая зависимость

$$\lambda(t) = k(M - i) = \lambda_i,$$

где: M – неизвестное первоначальное число ошибок; i – число обнаруженных ошибок, зависящих от времени t; k – константа.

Значения неизвестных параметров k и M может быть оценено на основе последовательности наблюдений интервалов между моментами обнаружения ошибок по методу максимального правдоподобия.

Рассмотрим модель Шика–Волвертона. Это модификация модели Джелинского-Моранды для случая возникновения на рассматриваемом интервале более одной ошибки. При этом считается, что исправление ошибок производится лишь после истечения интервала времени, на котором они возникли. В основе модели Шика–

Волвертона лежит предположение, согласно которому частота ошибок пропорциональна не только количеству ошибок в программах, но и времени тестирования, т.е. вероятность обнаружения ошибок с течением времени возрастает. Интенсивность обнаружения ошибок λ_i предполагается постоянной в течение интервала времени t_i и пропорциональна числу ошибок, оставшихся в программе по истечении (i–1)-го интервала; но она пропорциональна также и суммарному времени, уже затраченному на тестирование:

$$\lambda_i = C(N - n_{i-1})\left(T_{i-1} + \frac{t_i}{2}\right),$$

где: C – коэффициент пропорциональности; N – число ошибок, первоначально присутствующих в программе; $n_{i\text{-}1}$– число ошибок, оставшихся в программе по истечении (i-1)-го интервала; $T_{i\text{-}1}$– суммарное время, затраченное на тестирование в течение (i–1) этапов; t_i – среднее время выполнения программы в текущем интервале.

Остальные расчеты аналогичны расчетам модели Джелинского–Моранды.

Представленные модели не учитывают повторяемость этапов разработки программного обеспечения. После этапа разработки и отладки идёт этап тестирования, и по его результатам принимается решение о готовности продукта или необходимости его доработки. Но на этапе доработки могут возникнуть изменения, которые могут повлиять на надёжность программного обеспечения.

Модифицированная модель Джелинского–Моранды. Предлагается, что при исправлении ошибок, обнаруженных на предыдущем этапе тестирования возможно, возникновение новых.

$$\lambda(t) = k(M + H - i) = \lambda_i,$$

где: H – количество ошибок, появившихся в результате исправления ошибок, обнаруженных на предыдущем этапе тестирования.

$$H = f \cdot g \cdot i \cdot \sum_j P_j,$$

где: i – число ошибок на предыдущем этапе; P_j– вероятность появления новой ошибки (P_1– вероятность появления одной ошибки, P_2 –вероятность появления второй ошибки и т.д.); f – коэффициент опыта команды; g – коэффициент готовности продукта (функция от времени)

Модифицированная модель Шика–Волвертона.

$$\lambda_i = C(N + H - n_{i-1})\left(T_{i-1} + \frac{t_i}{2}\right),$$

где: H – количество ошибок, появившихся в результате исправления ошибок, обнаруженных на предыдущем этапе тестирования.

$$H = f \cdot g \cdot i \cdot \sum_j P_j,$$

где: i – число ошибок на предыдущем этапе; Pj – вероятность появления новой ошибки (P_1– вероятность появления одной ошибки, P_2 – вероятность появления второй

ошибки и т.д.); f – коэффициент опыта команды; g – коэффициент готовности продукта (функция от времени).

Были проведены исследования для определения вероятности появления новой ошибки. Исследования проводились в течение года на двух независимых группах разработчиков.

В результате проведённых исследований опытным путём было доказано, что соотношение количества ошибок, возникших в результате исправления, ранее обнаруженных, константно на небольшом промежутке времени

$$\frac{a}{i} = const,$$

где: a – количество ошибок, возникших в результате исправления, ранее обнаруженных;

i – общее число исправляемых ошибок.

Из этого утверждения и полученных данных следует, что соотношение количества единичных ошибок к общему числу исправляемых ошибок константно. Аналогично для двоичных, троичных и более ошибок

$$P_j = \frac{a_j}{i},$$

где: a_j – количество единичных ошибок, возникших в результате исправления, ранее обнаруженных;

i – общее число, исправляемых ошибок;

P_j – вероятность возникновения ошибки.

Согласно полученным соотношениям предлагается методика получения вероятностей возникновения ошибок.

Согласно теории разработки программного обеспечения, большие задачи необходимо решать с использованием метода декомпозиции. Учитывая это, можно создать специальную тестовую задачу для группы разработчиков, которая даст возможность оценить качество безошибочного программирования каждого разработчика из команды. Предлагается создать такую тестовую задачу, при которой будут затрагиваться задания, аналогичные задачам, которые возникают при выполнении требований целевой задачи. Тестовая задача должна включать в себя как разработку отдельных модулей, так и организацию их взаимодействия между собой. Задача должна решаться в два этапа. На первом этапе идёт разработка ПО, на втором – исправление ошибок. По результатам выполнения тестового задания можно получить вероятности возникновения единичных, двоичных и более ошибок.

Так как всё же существует некоторая флуктуация в соотношениях количества ошибок, появившихся в результате исправления ошибок, к общему числу исправляемых ошибок, то вводится «коэффициент недоверия». Учитывая, что мы оцениваем интенсивность отказов не точно, а только её верхнюю границу, предлагается к соотношению количества ошибок, появившихся в результате исправления ошибок, к общему числу исправляемых ошибок прибавлять коэффициент недоверия. Коэффициент недоверия зависит от величины отклонения соотношений ошибок к среднему значению. Таким образом, вероятность возникновения ошибки стоит рассчитывать по формуле:

$$P_j = \frac{a_j}{i} + x,$$

где: a_j – количество единичных ошибок, возникших в результате исправления, ранее обнаруженных;

i – общее число, исправляемых ошибок;

P_j – вероятность возникновения ошибки;

x – коэффициент недоверия.

Коэффициент недоверия определяется по методу максимального правдоподобия.

Так как со временем навыки разработки ПО членами команды разработчиков увеличиваются, то рекомендуется пересчитывать вероятности возникновения ошибок. Исходными данными для этого могут служить значения, полученные на предыдущем этапе разработки.

Определение коэффициента опыта команды f. Если команда не меняется на всём протяжении разработки, то коэффициент опыта команды можно приравнять 1. В случае изменения состава команды надо решить тестовую задачу с участием нового члена команды.

В статье рассмотрены модели надёжности программного обеспечения Джелинского–Моранды и Шика–Волвертона. Описан их недостаток. Предложены варианты модификации. Приведены исследования вероятностей возникновения ошибок при исправлении ранее обнаруженых.

СПИСОК ЛИТЕРАТУРЫ:

1. Азымшин И.М., Чуканов В.О. Анализ безопасности программного обеспечения // Безопасность информационных технологий. 2014. № 1. С. 45-47.
2. Чуканов В.О. Надежность программного обеспечения и аппаратных средств систем передачи данных атомных электростанций: Учебное пособие. М.: МИФИ, 2008.
3. Гуров В.В., Чуканов В.О. Основы теории и организации ЭВМ. М.: БИНОМ. Лаборатория знаний, 2012.
4. Александрович А.Е., Бородакий Ю.В., Чуканов В.О. Проектирование высоконадежных информационно-вычислительных систем. М.: Радио и связь, 2004.
5. Половко А. М., Гуров С. В. Основы теории надёжности. СПб.: БХВ-Петербург, 2006.

REFERENCES:

1. Azymshin I.M., Chukanov V.O. Analysis of safety of software // Bezopasnost Informatsionnykh Tekhnology. 2014. № 1. P. 45-47.
2. Chukanov V.O. Nadezhnost programmnogo obespecheniya i apparatnykh sredstv sistem peredachi dannykh atomnykh elektrostantsy: Uchebnoye posobiye. M.: MEPhI, 2008.
3. Gurov V.V., Chukanov V.O. Osnovy teorii i organizatsii EVM. M.: BINOM. Laboratoriya znany, 2012.
4. Aleksandrovich A. Ye., Borodaky Yu. V., Chukanov V. O. Proyektirovaniye vysokonadezhnykh informatsionno-vychislitelnykh sistem. M.: Radio i svyaz, 2004.
5. Polovko A. M., Gurov S. V. Osnovy teorii nadyozhnosti. SPb.: BKhV-Peterburg, 2006