ИССЛЕДОВАНИЕ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ ПРОГНОЗИРОВАНИЯ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Хунов Т. Х., Медведев Д.В., Полесский С. Н.
Москва, НИУ ВШЭ

В работе приведено исследование оценки надежности программных средств на примере трех математических моделей.

Estimation of reliability of software by mathematical models

The 3 models of software reliability estimating are researched in this work: The Jelinski-Moranda model, the Musa model and the Exponential model. Based on the Musa model calculation of reliability given with failures of software is done.

Данное научное исследование (№ проекта 14-05-0038) выполнено при поддержке Программы «Научный фонд НИУ ВШЭ» в 2014 г.

Прогнозирование надежности программного обеспечения осуществляется при помощи математических моделей, в основу которых заложены основы теории вероятности и математической статистики [1].

В основу математических моделях производится поиск количества ошибок, которые остаются в программе. Кроме того, с помощью этих моделей можно найти надежность программы (работа программы без выявления ошибок с течением времени, заданного до начала функционирования программы).

На сегодняшний день можно выделить 10 самых известных моделей надежности программного обеспечения (см. рис. 1) [1]. В работе эти модели сгруппированы по признакам.

- время нахождения ошибок в ПО;
- сложность самой программы;
- разметка ошибок (преждевременное внесение в программное обеспечение ошибок, которые известны заранее);
- структура входных данных;
- текстовая структура программы.

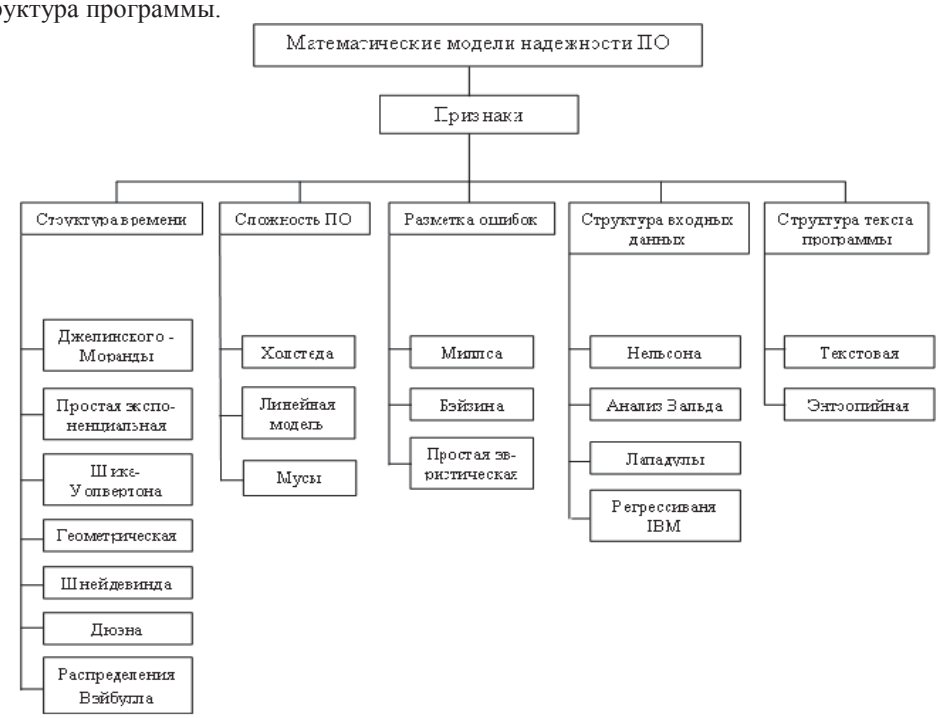

Рис. 1. Классификация моделей надежности ПО

В работе были рассмотрены три моделей оценки надежности программных средств: модель Джелински-Моранды, модель Мусы и простая экспоненциальная модель.

Модель Джелински-Моранды

Модель Джелински-Моранды предложена в 1996г. Иначе данная модель называется моделью роста надежности. Она основана на предположении об экспоненциальной зависимости плотности вероятности интервалов времени между проявлением ошибок от интенсивности ошибок. Кроме того, в модели полагается, что интенсивность ошибок на каждом случайном интервале времени линейно зависит от количества оставшихся в программе ошибок.

Эта модель является одной из наиболее распространенных моделей надежности программного обеспечения.

Модель во многом связана с теорией надежности аппаратуры.

Одним из способов оценки надежности является наблюдение за программой в течение какого-то промежутка времени и изображение на графике значений между ошибками, появляющимися последовательно. Рисунок 2 иллюстрирует повышение надежности программного обеспечения (R_1, R_2, R_3 являются функциями надежности, т. е. вероятность того, что в программе не будет ни одной ошибки или ошибка не будет выявлена на промежутке времени от *0* до *t*), если исправить в нем ошибки.

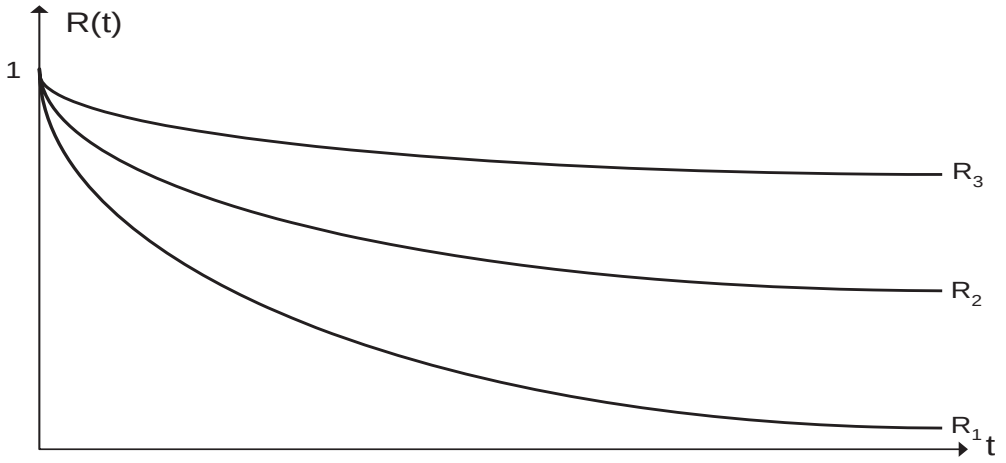

Рис. 2. Увеличение надежности ПО при исправлении в нем ошибок.

Для того чтобы разработать модель нужно ввести несколько предположений:

- $\lambda_i = Z(\mathrm{t})$ будет постоянным до того, как ошибки будут обнаружены и исправлены;
- $\lambda_i = Z(\mathrm{t}) = K \cdot (\mathrm{N} - \mathrm{i})$ - т.е. $Z(\mathrm{t})$ прямо пропорционально количеству оставшихся в программе ошибок, где *N* - неопределенная начальное количество ошибок, *i* - количество обнаруженных в эксплуатации программы ошибок, *K* – какая-то константа.

После того как какое-то количество ошибок уже выявлено, Неизвестные *N* и *K* можно будет оценивать. Допустим, что выявлено *n* ошибок, а $x[1], x[2], \ldots, x[n]$ - промежутки времени между обнаруженными ошибками. Тогда, исходя из того что $Z(\mathrm{t})$ постоянно между ошибками, вероятностная плотность для $x[\mathrm{i}]$ будет равна:

$$p\left(x[i]\right) = K \cdot \left(N - i\right) \cdot \exp\left\{-K \cdot \left(N - i\right) \cdot x[i]\right\}.$$

Основным результатом *N* является оценка конечного количества ошибок. Если знать значение *K,* можно использовать это уравнение, чтобы предсказать время до выявления *(n + 1)*-й ошибки и следующих ошибок.

Этот случай является частным случаем Шуманской модели. Допустим, что промежутки времени отладки между выявлениями 2-х ошибок имеет зависимость, которая называется экспоненциальной, и частота этих ошибок прямо пропорциональна количеству ошибок, которые еще не обнаружены. Итак,

функция плотности распределения времени выявления *i*-ой ошибки, которая считается с момента выявления *(i-1)*-ой ошибки выглядит следующим образом $\rho(t_i) = \lambda_i \cdot \exp(-\lambda_i \cdot t_i)$, где $\lambda_i = \Phi(N - i + 1)$ и N - это количество ошибок, присутствующих изначально в программном обеспечении.

Это модель является одной из самых первых моделей. В основу модели были положены несколько допущений:

- интенсивность выявления ошибок $R(t)$ прямо пропорциональна текущему количеству ошибок в данной программе;
- все ошибки могут появиться с одинаковой вероятностью;
- время до следующей ошибки распределено экспоненциально;
- корректировка ошибок осуществляется без внесения в программу других ошибок;
- $R(t) = const$ между 2-мя соседними ошибками [2].

Модель Мусы

Модель была разработана в середине 1970-х годов и является одной из самых первых моделей прогнозирования надежности. Эта модель прогнозирует начальную интенсивность отказов ПО в момент, когда начинается тестирование ПО(т. е. когда время t = 0). Формула прогнозирования выглядит следующим образом [3]:

$$\lambda_0 = k \cdot p \cdot w_0,$$

где λ_0 - начальная интенсивность отказов.

Например, 100 строк (SLOC) на языке Fortran, где средний уровень выполнения 150 строк в секунду, имеет предсказуемую интенсивность отказов. Когда начинается тестовая система $\lambda_0 = k \cdot p \cdot w_0 = 4.2e^{-7} \cdot (150 \div 100 \div 3) \cdot (6 \div 1000) = 1.26e^{-9}$ ошибки в секунду (или 1 ошибка на $7.9365 \cdot 10^8$ секунд, что эквивалентно 1-й ошибке за 25,17 лет).

Важно отметить, что время выполнения операций не календарное. Так как модели надежности рассчитываются по календарному времени, модель Мусы невозможно использовать в разработке общей оценки надежности системы, если только не предположить, что календарное время и время выполнения являются одинаковыми [3].

Таблица 1. Термины, использующиеся в модели Мусы

Символы	Значения	Величина
k	Константа динамической структуры программы	$4,2 \cdot 10^{-7}$
p	Оценка числа исполнений в единицу времени	p = r/SLOC/ER
r	Средняя скорость выполнения операций (определяется производителем)	Константа
SLOC	Исходные строки кода (не включая комментарии, т.е. размер программ)	
ER	Коэффициент расширения (зависит от языка программирования)	Assembler - 1.0; C - 2.5; COBAL, FORTRAN – 3.0
w_0	Оценка первоначальных ошибок в программе	Может быть рассчитана как $w_0 = N \cdot B$ (по умолчанию 6/1000 SLOC)
N	Общее количество присущих ошибок	Рассчитываются на основании суждений или прошлого опыта
B	Ошибки, превращающиеся в сбои. Ошибки, неисправленные до поставки продукта.	Предположим B = 0,95 или 95 % незамеченные при доставке становятся сбоями после доставки.

Модель Мусы является одной из самых известных моделей оценки надежности ПО. На основе этой модели сделаем расчет надежности с учетом интенсивности отказов ПО, а так же печатной платы. Проведем расчет надежности ПО с использованием методики справочника MIL HDBK 217Plus [4].

Основная модель для расчета надежности ПО:

$$\lambda_{sw} = \left(\frac{F_{t_{i-1}} - F_{t_i}}{730} \right) \cdot (DC \cdot FL \cdot FA \cdot AS) \cdot 10^6,$$

где λ_{sw} - интенсивность отказов ПО в месяц (в отказах за 1×10^6 календарных часов);

- F_{t_i} - количество отказов, ожидаемых за время(t_i);

Формула, по которой рассчитывается F_{t_i} :

$$F_{t_i} = F_0 \, e^{-k t_i}$$

где: F_0 – начальная интенсивность отказов $F_0 = KSLOC \cdot FD$

k – темп роста отказов $k = \dfrac{\ln\left(\dfrac{1}{DSL}\right)}{t_s}$

t_i – время после релиза ПО (в месяцах)

Таблица 2. Параметры, используемые в модели Мусы

Параметр Символ	Имя	Описание	Недостаток
KSLOC	Линии исходного кода (в тысячах)	Строки исходного кода (в тысячах),не включая комментарии, т.е. размер программ	Нет
FD	Неисправность компактности	Исходное качество как измеряется компактность неисправностей	
FL	Неисправность задержки	Среднее число раз сбой, как ожидается, будет повторяться до его основной ошибки	2.0 (безразмерное число)
FA	Неисправность активации	Фракция (в десятичной форме) выставления активизации разломов популяции	1.0 (100%)
AS	Средний % трудности	Фракция (в десятичной форме) неисправностей, которые разрушительные, или критические, заказчику	0.5 (50%)
ts	Время стабилизации		48 (месяцев) для начальной версии программного обеспечения. Тем не менее, это значение должно быть изменено на 24 (месяцев) для последующих версий программного обеспечения.
DSL	Уровень дефекта стабилизации	Уровень, на котором интенсивность отказов программного обеспечения стабилизируют относительно *F0*	
DC	Рабочий цикл	Доля календарного времени, при котором программное обеспечение находится в эксплуатации (в десятичной форме)	Операционный профиль рабочий цикл

Расчет начнем с определения *KSLOC* (число строк в программе, за исключением комментариев). Листинг пода программы показан в приложении. При учете количества строк в программе не учитываются комментарии к коду программы.

KSLOC = 178

Далее определим начальный уровень неисправностей при разработке (*FD*).Условимся, что опыта разработки программного обеспечения нет, тогда:

- *FD = 4.0*;

Определим F_0:

$$F_0 = KSLOC \cdot FD = 178 \cdot 4,0 = 712;$$

Определим *k:*

DSL = 0.10 (см. табл. 3), $t_s = 48 месяцев$ (см. табл. 2)

$$k = \frac{\ln\left(\frac{1}{DSL}\right)}{t_s} = \frac{\ln\left(\frac{1}{0,1}\right)}{48} = 0,048$$

Найдем F_{t_i} :

$$F_{t_i} = F_0 e^{-k t_i} = 712 \cdot e^{-(0,048 \cdot 12)} = 400,245$$

Найдем $F_{t_i - 1}$:

$$F_{t_i - 1} = F_0 e^{-(k t_i - 1)} = 712 \cdot e^{-(0,048 \cdot 11)} = 419,9$$

Возьмем значения FL, FA, AS из таблицы 1. *DC* примем равным 2.

Тогда:

$$\lambda_{sw} = \left(\frac{F_{t_{i-1}} - F_{t_i}}{730}\right) \cdot \left(DC \cdot FL \cdot FA \cdot AS\right) \cdot 10^6 = \left(\frac{419,9 - 400,245}{730}\right)\left(1 \cdot 2 \cdot 1 \cdot 0,5\right) \cdot 10^{-6} = 0,027 \cdot 10^{-6}$$

Таблица 3. Значения по умолчанию дефекта компактности и дефекта стабилизация уровня

SEI CMM уровень	Первоначальный дизайн Дефект компактности (FD) (Дефекты в 1000 строк. Код для всех степеней тяжести)	Уровень дефекта стабилизации(DSL)
5	0.5	0.01
4	1.0	0.03
3	2.0	0.05
2	3.0	0.07
1	5.0	0.10
Не в рейтинге	6.0	Не расчетное

Простая экспоненциальная модель.

Отличительной чертой простой экспоненциальной модели от модели Джелински-Моранды заключается в том, что $R(t) \neq const$. [5]

Допустим $N(t)$ – количество шибок, выявленных к моменту времени t и функция риска прямо пропорциональна количеству ошибок, не обнаруженных к моменту t в программе :

$$R(t) = K \cdot \left(N(0) - N(t)\right).$$

Если продифференцировать обе части уравнения по времени:

$$\frac{dR(t)}{dt} = -K \cdot \frac{dN(t)}{dt}.$$

И если учитывать, что $dN(t)/dt$ есть $R(t)$ (количество ошибок, выявленных в единицу времени), мы получим дифференциальное уравнение:

$\frac{dR(t)}{dt} + K \cdot R(t) = 0$ с начальными условиями $N(0) = 0$, $R(0) = K \cdot N_0$.

Если решить это уравнение, получится следующая функция:

$$R(t) = K \cdot N_0 \cdot \exp(-K \cdot t), \quad (1)$$

Коэффициенты K и N_0 рассчитываются по следующим формулам:

$$K = \frac{\frac{1}{n}\left(\sum_{i=1}^{n} \ln R(t_i)\right) \cdot \sum_{i=1}^{n} t_i - \sum_{i=1}^{n} t_i \cdot \ln R(t_i)}{\sum_{i=1}^{n} t_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} t_i\right)^2}$$

$$N_0 = \frac{\exp(a)}{K}$$

Используя уравнение (1) мы сможем определить время, которое необходимо для уменьшения интенсивности появления ошибок с $R_1(t)$ до $R_2(t)$:

$$t_{12} = \frac{1}{K} \ln \frac{R_1}{R_2}.$$

Выводы

Модели надежности программных средств являются одним из средств оценки и повышения ПС. В настоящее время существует множество моделей оценки надежности ПО. Одними из самых известных моделей являются: модель Джелински-Моранды, модель Мусы и простая экспоненциальная модель. По этим моделям производится расчет показателей надежности (вероятность безотказной работы, вероятность исправления всех ошибок, средняя наработка отказа).

Литература

1. Липаев В.В. Надежность программных средств. - М.: СИНТЕГ, 1998.
2. ГОСТ 28195-89. Оценка качества программных средств. Общие положения. - М.: Госком. СССР по стандартам.
3. Military Handbook. Electronic reliability design handbook, 1998.
4. Handbook of 217 plus. Reliability prediction models, 2006.
5. Гуляев В.А., Коростиль Ю.М. Анализ и исследование методов оценки и увеличения надежности программ. – Киев, 1990.

ТРЕБОВАНИЯ К ИМИТАТОРУ КАЧКИ ДЛЯ КОРАБЕЛЬНОЙ СИСТЕМЫ СПУТНИКОВОЙ ПОСАДКИ.

Теличкань В.С., *Увайсов С.У., Смирнов Д.О., Щеткова Т.А.
*ОАО «МКБ «Компас», * НИУ ВШЭ*