Интернет-журнал «Науковедение» ISSN 2223-5167 https://naukovedenie.ru/

Том 9, №6 (2017) https://naukovedenie.ru/vol9-6.php

URL статьи: https://naukovedenie.ru/PDF/09TVN617.pdf

Статья опубликована 18.12.2017

Ссылка для цитирования этой статьи:
Бурзуева Н.Н., Мостовой Я.А. Анализ надежности среды разработки android studio // Интернет-журнал «НАУКОВЕДЕНИЕ» Том 9, №6 (2017) https://naukovedenie.ru/PDF/09TVN617.pdf (доступ свободный). Загл. с экрана. Яз. рус., англ.

УДК 004.052.4

Бурзуева Наталья Новрузовна
ФГБОУ ВО «Поволжский государственный университет телекоммуникаций и информатики», Россия, Самара
Аспирант
E-mail: nburzueva@gmail.com

Мостовой Яков Анатольевич
ФГБОУ ВО «Поволжский государственный университет телекоммуникаций и информатики», Россия, Самара
Доктор технических наук, профессор
E-mail: jakob.mostovoi@yandex.ru

Анализ надежности среды разработки android studio

Аннотация. В данной статье рассмотрены две взаимодополняющие друг друга модели проявления ошибок (изменений) в программном обеспечении: дискретная и непрерывная, опирающиеся на параметры, вытекающие из гипотезы «Джелинского-Моранды», определенные по статистике проявления ошибок программного обеспечения во времени. Получено выражение, позволяющее определить вероятность отсутствия ошибок в программном обеспечении через заданное время.

Предлагается считать, что любое проведённое разработчиком формально зарегистрированное исправление кода является устранением ошибки или уязвимости, наличие которой также в настоящее время является ошибкой (изменением) либо программной, либо алгоритмической, которые иногда «маскируются» как внесение улучшений. При таком подходе появляется возможность оценить надежность программного обеспечения в случае, когда имеется статистика его изменений.

Вопросы расчёта показателей надёжности программного обеспечения рассмотрены в статье применительно к среде разработки Android Studio. Для проведения расчетов была собрана статистика возникновения ошибок за семь месяцев с критическим и высоким приоритетом по интегрированной среде разработки Android Studio по результатам сайта технической поддержки Android Open Source Project – Issue Tracker и определен с вероятностью 0,9 интервал времени через который в программном обеспечении будут отсутствовать ошибки (изменения).

Ключевые слова: надежность программного обеспечения; дискретная и непрерывная модели; марковский случайный процесс; модель Джелинского-Моранды; метод наименьших квадратов; Android Studio; техническая поддержка

Введение

В работах [1, 2, 4, 5] рассмотрены различные показатели надёжности программного обеспечения (далее ПО) и подходы к их определению и отмечается, что ни один из подходов не получил широкого распространения. Одна из причин этого связана с неразвитостью методов расчёта, другая – с тем, что факты и причины проявления ошибок в ПО часто неохотно публикуются и указываются разработчиком. Часто эти факты и причины объявляются разработчиком связанными с проведением улучшений по требованиям заказчика, что затрудняет проведение анализа.

С другой стороны, определение фактической надёжности конкретного ПО представляется важным, хотя бы с точки зрения прогноза её дальнейшего изменения.

В этой ситуации предлагается считать, что любое проведённое разработчиком формально зарегистрированное исправление кода (далее изменение) является устранением ошибки или уязвимости, наличие которой также в настоящее время является ошибкой либо программной, либо алгоритмической, которые иногда «маскируются» как внесение улучшений. При таком подходе появляется возможность оценить надежность ПО в случае, когда имеется статистика его изменений.

Вопросы расчёта показателей надёжности программного обеспечения рассмотрены в статье применительно к среде разработки Android Studio. Android – это портативная (сетевая) операционная система для коммуникаторов, планшетных компьютеров, электронных книжек, цифровых проигрывателей, наручных часов, нетбуков и смартбуков, основанная на ядре Linux. Android Studio, основанная на программном обеспечении IntelliJ IDEA от компании JetBrains, официальное средство разработки Android приложений [10]. Данная среда разработки доступна для Windows, OS X и Linux [11].

Для проведения расчетов для определения количества оставшихся ошибок и для определения времени полного устранения ошибок с заданной вероятностью интегрированной среды разработки Android Studio была собрана статистика проявления ошибок с высоким и критическим приоритетом, находящаяся в открытом доступе на сайте https://code.google.com/p/android/ за период времени 7 месяцев [12].

При этом нами предложено использовать две модели проявления ошибок непрерывную и дискретную. Обе модели опираются на гипотезу Джелинского-Моранды [1, 5]. Непрерывная модель «заточена» на наблюдённое на некотором интервале времени количестве ошибок в ПО, т. е. непосредственно на результаты наблюдений. По ней удобно определять параметры модели надёжности, вытекающие из гипотезы Джелинского-Моранды. Однако, на выходе у непрерывной модели – число оставшихся в ПО ошибок, которое принимает с течением времени всё меньшее и меньшее значение, но никогда не достигает 0. Это затрудняет интерпретацию результатов расчётов.

Поэтому рассмотрена также дискретная модель проявления ошибок программного обеспечения, опирающаяся на марковский случайный процесс типа Юла-Фарри с переменной интенсивностью переходов между состояниями процесса с учётом тех же параметров Джелинского-Моранды. Эта дискретная модель позволяет, рассматривая случайный динамический процесс проявления ошибок при отладке или эксплуатации программного обеспечения, определить показатели его надёжности за требуемое время. При этом обработка статистики по проявившимся с течением времени ошибкам проводится процедурой метода наименьших квадратов [3, 7] в рамках непрерывной модели.

Методы исследования. Дискретная модель проявления ошибок в ПО

Свяжем состояния дискретного марковского случайного процесса проявлений ошибок в ПО с количеством обнаруженных ошибок [7]. Состоянию S_0 соответствует 0 обнаруженных ошибок – это начальное состояние нашего дискретного марковского процесса. Состоянию S_1 соответствует одна проявившаяся ошибка, S_2-2, S_3-3, S_n-n ошибок и состоянию S_{N_0} соответствует N_0 ошибок. Пусть всего в ПО N_0 ошибок. С течением времени все они проявятся и процесс завершится (см. рис. 1). С каждым состоянием процесса свяжем вероятность пребывания в этом состоянии. $P_i(t)$. При этом в дальнейшем t в обозначениях вероятностей мы будем опускать, подразумевая, что P – это функции времени. Данный случайный процесс с «линейчатым» графом характерного вида носит название процесса «гибели или размножения» и широко применяется в теории надежности, теории массового обслуживания [5]. Имея граф состояний процесса можно расписать Уравнения Колмогорова.

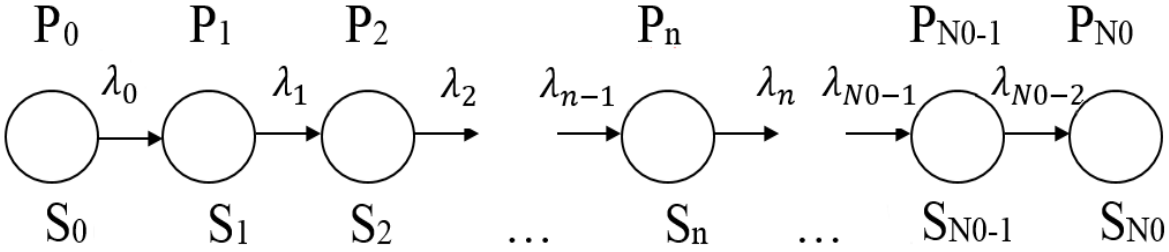

Рисунок 1. *Граф состояний процесса «гибели» (составлено: Мостовой Я. А.)*

Система уравнений Колмогорова для данного случайного процесса имеет вид:

$$\begin{gathered}\frac{\mathrm{dP_0}}{\mathrm{dt}} = -\lambda \mathrm{P_0} + \mathrm{P_{0(0)}} \\ \frac{dP_1}{dt} = -\lambda_1 P_1 + \lambda_0 P_{0(0)} \\ \cdots\cdots\cdots\cdots \\ \frac{dP_n}{dt} = -\lambda_n P_n + \lambda_{n-1} P_{n-1} \\ \cdots\cdots\cdots\cdots \\ \frac{dP_{N_0}}{dt} = -\lambda_{N_0-1} + P_{N_0-1}\end{gathered} \tag{1}$$

Начальные условия для этой системы уравнений:

$$\mathrm{P_{0(0)}} = 1,\ \mathrm{P_1} = \mathrm{P_2} \ \ldots\ \mathrm{P_{N_0}} = 0 \tag{2}$$

Нормирующее условие:

$$\sum_{0}^{\mathrm{N_0}} \mathrm{P_n} = 1 \tag{3}$$

Однако, для случайного процесса проявления ошибок в чистом виде процесс «гибели» или размножения с постоянной интенсивностью проявления ошибок не совсем подходит.

Практика использования ПО и физические соображения говорят, что с течением времени по мере обнаружения все большего числа ошибок в ПО и их исправления интенсивность проявления оставшихся ошибок уменьшается т. е. λ с течением времени уменьшается. Гипотеза Джелинского-Моранды позволяет учесть данный факт, связывая, однако, переменность интенсивности проявления ошибок не со временем, а с состояниями

процесса – числом оставшихся ошибок. Это позволяет нам оставаться в рамках марковских случайных процессов.

Джелинский и Моранда сделали предположение – гипотезу [1]: интенсивность проявления ошибок пропорциональна числу оставшихся ошибок.

$$\lambda_n = (N_0 - n) * k, \quad (4)$$

где: k – некоторый коэффициент пропорциональности;

n – количество проявившихся ошибок;

N_0 – начальное количество ошибок;

(N_0 – n) – количество оставшихся ошибок.

Запишем уравнения Колмогорова для случайного процесса проявления ошибок и подставим в них значения интенсивности проявления ошибок в соответствии с гипотезой Джелинского-Моранды.

$$\lambda_0 = N_0 k, \lambda_1 = (N_0 - 1)k, .. \lambda_n = (N_0 - n)k \quad (5)$$

Полученный процесс с зависящей от состояния процесса интенсивностью перехода λ_n, рассмотренный нами, является случайным процессом типа Юла-Фарри.

Для процесса типа Юла-Фарри применение преобразования Лапласа позволяет получить изображения вероятностей пребывания процесса в интересующих состояниях. В нашем случае интерес представляет вероятность состояния, когда все имеющиеся первоначально N_0 ошибки проявились. Применим последовательно преобразование Лапласа к ранее записанной системе уравнений Колмогорова с учетом начальных условий.

$$\begin{gathered} \mathbf{P_0(S)=\frac{1}{S + k * N_0}} \\ \mathbf{P_1(S)=\frac{P_{n-1}(S)*k(N_0-n+1)}{S + k * (N_0-n)}} \\ \mathbf{\ldots\ldots} \\ \mathbf{P_n(S)=\frac{P_{n-1}(S)*k(N_0-n+1)}{S + k * (N_0-n)}} \\ \mathbf{\ldots\ldots} \\ \mathbf{P_{N_0}(S)=\frac{PN_{0-1}(S)*k}{S}} \end{gathered} \quad (6)$$

Последовательно подставляя в изображение для каждого $P_n(s)$ выражение для $P_{n-1}(s)$, имеем:

$$\mathbf{P_n(S)=\frac{k^n*N_0*(N_0-1)*N_0-2\cdots*(N_0-n+1)}{(S+k*N_0)*\big((S+k*(N_0-1)\big)*(S+k*N_0-2))*\cdots*(S+k*(N_0-n))}} \quad (7)$$

Здесь: P_n – вероятность проявления n ошибок.

Разделим и числитель, и знаменатель на k^n. В случае, когда $n = N_0$ т. е. когда все ошибки обнаружены, имеем изображение для ВБР.

$$\mathbf{P_{N_0}(S)=\frac{N_0!}{s(\frac{1}{k}s+1)(\frac{1}{k}s+2)(\frac{1}{k}s+3)\ldots(\frac{1}{k}s+N_0)}} \quad (8)$$

В достаточно полных таблицах преобразования Лапласа есть подобные изображения и оригиналы. Последнее изображение приведено к табличному виду [4].

Применим обратное преобразование Лапласа.

$$\mathbf{P_{N_0}(t) = (1 - e^{-kt})^{N_0}} \quad (9)$$

Данная формула определяет вероятность обнаружения (проявления) всех N_0 ошибок за время t.

В данной формуле все прекрасно, но как определить N_0 и k? N_0 и k неизвестны и более того они индивидуальны для каждого ПО, для каждого программиста и не могут иметь общего значения.

Методы исследования. Непрерывная модель проявления ошибок в ПО

Модель изменения надежности программного обеспечения, основанная на гипотезе Джелинского-Моранды позволяет, основываясь на наблюдениях за процессом проявления ошибок в программном обеспечении, определить характеристику надежности программного обеспечения – интенсивность проявления ошибок λ, а точнее начальное количество ошибок N_0 и параметр k, определяющий скорость проявления ошибок.

Решение дифференциального уравнения первого порядка относительно числа проявившихся ошибок n, вытекающего из записи (4).

$$\mathbf{n = N_0 * \left(1 - e^{-k*t}\right)} \quad (10)$$

Запишем выражение для количества ошибок, проявившихся на интервале Δn_i, полагая равномерным шаг наблюдений за проявлениями ошибок Δt.

$$\Delta n_i = N_0 * \left(1\text{-}e^{-k*t_i}\right)\text{-}N_0 * \left(1\text{-}e^{-k*t_{i-1}}\right) = N_0 * e^{-k*t_i}(-1 + e^{k*\Delta t}) \quad (11)$$

Здесь: Δn_i – приращение количества обнаруженных ошибок на интервале Δt.

Пусть шаг Δt можно выбрать так, чтобы произведение $k*\Delta t < 1$.

Тогда, разлагая exp функцию в ряд и пренебрегая членами выше 1-ой степени $k*\Delta t$, как малыми, имеем:

$$\exp(x) = 1 - \frac{x}{1!} + \frac{x^2}{2!} \cdots$$
$$\Delta n_i \approx N_0 * e^{-kt_i} * (1 + k * \Delta t - 1) = N_0 * k * \Delta t * e^{-k*t_i} \quad (12)$$

Для определения методом наименьших квадратов параметров N_0 и k запишем сумму квадратов невязок:

$$S = \sum_{i=1}^{r} (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i})^2 \quad (13)$$

Найдем значение N_0 и k, при которых эта сумма имеет минимум. Для этого приравняем нулю производные суммы по искомым параметрам.

$$\frac{\partial S}{\partial N_0} = \sum_{i=1}^{r} 2 * (\Delta n_i - N_0 * k\Delta t * e^{-kt_i}) * (-k * \Delta t * e^{-kt_i}) = 0$$
$$\sum_{i=1}^{r} (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i}) e^{-kt_i} = 0 \quad (14)$$

$$\frac{\partial S}{\partial k} = \sum_{i=1}^{r} 2 * (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i}) * (-N_0 \Delta t * e^{-kt_i} + N_0 * kt_i * \Delta t e^{-kt_i}) = 0$$

$$\sum_{i=1}^{r} (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i}) * e^{-kt_i} (k * t_i - 1) = 0$$

Окончательно запишем полученную систему 2-х уравнений с двумя неизвестными N_0 и k.

$$\sum_{i=1}^{r} (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i}) * t_i * e^{-kt_i} = 0$$

$$\sum_{i=1}^{r} (\Delta n_i - N_0 * k * \Delta t * e^{-kt_i}) * e^{-kt_i} = 0$$

Выразим из первого уравнения N_0, раскрывая скобки и вынося постоянные, не зависящие от индекса i, за знак суммы:

$$\sum_{i=1}^{r} \Delta n_i * t_i * e^{-k*t_i} - N_0 * k * \Delta t * \sum_{i=1}^{r} (e^{-k*t_i} * t_i) = 0 \quad (15)$$

$$N_0 = \frac{\sum_{i=1}^{r} (\Delta n_i * t_i * e^{-k*t_i})}{k * \Delta t * \sum_{i=1}^{r} (e^{-2*k*t_i} * t_i)} \quad (16)$$

Подставляя это выражение во второе уравнение, после преобразований имеем:

$$\sum_{i=1}^{r} (\Delta n_i * e^{-k*t_i}) = \frac{\sum_{i}^{r} \Delta n_i * t_i * e^{-k*t_i}}{\sum_{i}^{r} t_i * e^{-2k*t_i}} * \sum_{i=1}^{r} e^{-2k*t_i} \quad (17)$$

Мы получили трансцендентные уравнение с одним неизвестным k, которое может быть решено численно.

Полученная непрерывная модель надёжности ПО может независимо и достаточно достоверно оценивать ход процесса повышения надёжности ПО в процессе его отладки или эксплуатации, который целесообразно прекращать, когда интенсивность проявления ошибок достигла через определенное время очень низкого значения. Однако, имеются два вопроса, которые затрудняют интерпретацию результатов и которые надо рассмотреть для того, чтобы определить момент достижения требуемой надёжности ПО.

Во-первых, рассмотренная модель аппроксимации количества ошибок экспонентой, – модель непрерывная и может показывать дробное число ошибок на заданный момент времени, в том числе и значение количества оставшихся ошибок меньше 1. Во-вторых, модель никогда не покажет точный нуль оставшихся ошибок. Количество оставшихся ошибок будет приближаться к нему бесконечно долго. Конечно, это не соответствует физическому смыслу и может считаться недостатком рассмотренной непрерывной модели изменения надежности программного обеспечения.

Выше нами рассмотрена дискретная модель с непрерывным временем для определения вероятности проявления n ошибок, построенная на базе случайного процесса «гибели», также с использованием параметров Джелинского-Моранды и была получена формула (9), определяющая вероятность обнаружения всех ошибок из N_0, имеющихся в ПО.

Задавшись вероятностью обнаружения всех ошибок в ПО, мы можем определить время, для которого данная вероятность будет достигнута. Это позволит нам аккуратнее ответить на

вопрос сколько времени надо продолжать отладку – столько, чтобы вероятность отсутствия ошибок была, например, 0,9 или любым другим заданным числом.

Результаты исследования

Для проведения расчетов была собрана статистика возникновения ошибок за 7 месяцев с критическим и высоким приоритетом по интегрированной среде разработки Android Studio по результатам сайта технической поддержки Android Open Source Project – Issue Tracker, которая находится по адресу https://code.google.com/p/android/.

Для нахождения коэффициента k была организована итерационная численная процедура вычисления левой и правой части равенства (17) при различных значениях k Процедура прекращается и k считается найденным, как только разница между вычисленной левой и правой частью равенства, которую обозначим ε, станет меньше заданного малого числа, например, ε = 0,000001.

На рис. 2 представлен график изменения функции:

$$\varepsilon(k) = \left(\sum_{i=1}^{r} e^{-2kt_i}\right) * \frac{\sum_{i=1}^{r} \Delta n_i * e^{-kt_i} * t_i}{\sum_{i=1}^{r} e^{-2kt_i} * t_i} - \sum_{i=1}^{r} \Delta n_i * e^{-kt_i},$$

которая для k = 0,19 практически равна 0.

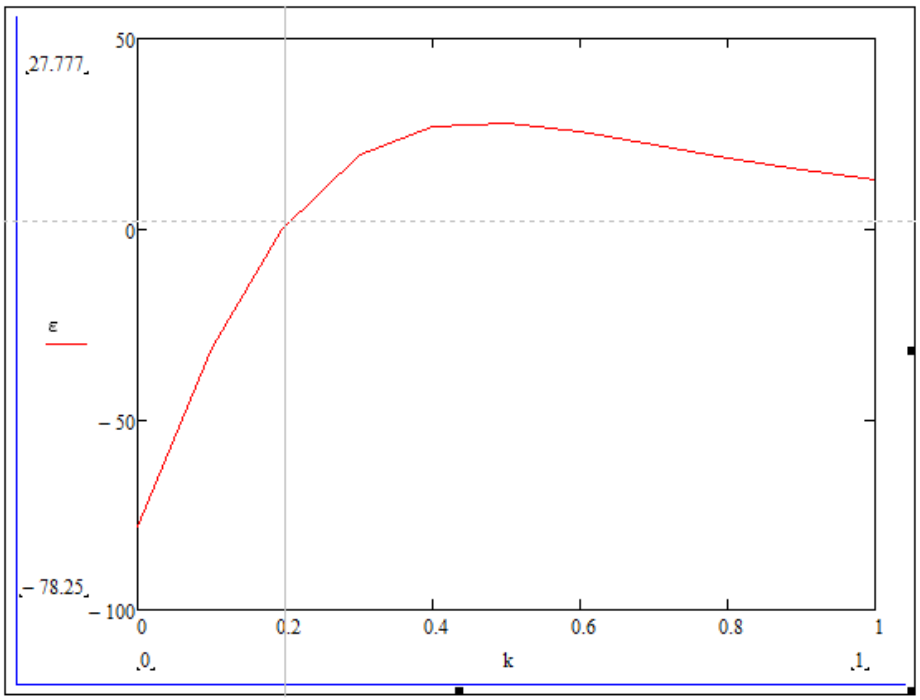

Рисунок 2. *График изменения функции ε(k) (составлено: Бурзуева Н. Н.)*

Полученное значение k подставим в формулу (15) для нахождения N_0. Получим результат N_0 = 626,5.

Проведён расчет сглаженных методом наименьших квадратов значений Δn_i по определенным N_0 и k и формуле (13) и построен совместный график изменений Δn_i по результатам наблюдений и сглаженных методом наименьших квадратов в функции t_i (рис. 3).

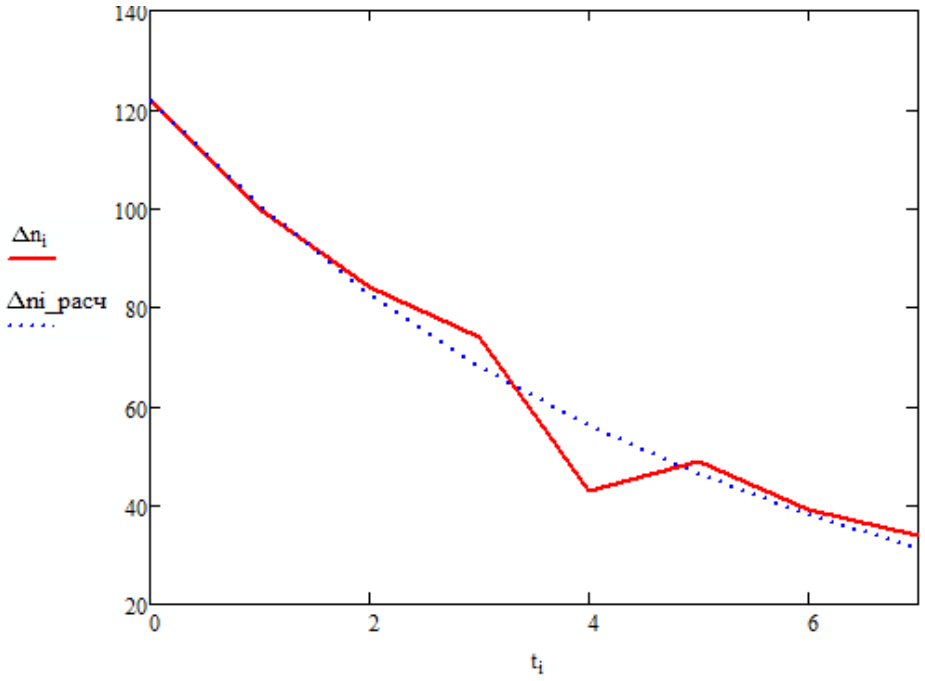

Рисунок 3. *Аппроксимация методом наименьших квадратов исходных данных (Δn_i) расчетными данными ($\Delta n_{i_{расчетное}}$) (составлено: Бурзуева Н. Н.)*

Задавшись вероятностью обнаружения всех ошибок в ПО P_{N_0} = 0,9 по формуле (9) определим время, за которое ПО Android Studio достигнет заданной надёжности. Это время составляет t = 44,628 месяца. Проведено предварительное сглаживание временного ряда «ошибок» методом среднего скользящего и затем сглаживание полученного результата методом экспоненциальной регрессии, однако, существенных улучшений по прогнозу устранения ошибок они не дали.

Выводы

1. Предлагается считать, что любое проведённое разработчиком формально зарегистрированное изменение кода является устранением ошибки или уязвимости, наличие которой также является ошибкой, либо программной, либо алгоритмической, которые иногда «маскируются» разработчиком, как внесение улучшений.

2. Рассмотрены совместно две модели для оценки надёжности ПО: непрерывная относительно числа проявившихся ошибок за время t и дискретная относительно вероятности проявления n ошибок на момент времени t. Обе модели опираются на одни и те же параметры, вытекающие из гипотезы Джелинского-Моранды, которые определены из статистики проявившихся ошибок по непрерывной модели методом наименьших квадратов.

3. Проведены расчеты по собранным статистическим данным технической поддержки. Изменение по времени количества проявившихся ошибок позволяет сделать вывод, что все ошибки с высоким и критическим приоритетом среды разработки Android Studio будут устранены через 44,628 месяца.

4. Проведено предварительное сглаживание временного ряда «ошибок» методом среднего скользящего и затем сглаживание полученного результата методом экспоненциальной регрессии, однако, существенных улучшений по прогнозу устранения ошибок они не дали.

ЛИТЕРАТУРА

1. Г. Майерс. Надёжность программного обеспечения. Москва: – Мир, 1980. – С. 360.
2. Романюк С. Г. Оценка надёжности ПО // Открытые системы №4, 1994 – С. 258.
3. Козлов, Д. И. Управление космическими аппаратами зондирования Земли. Компьютерные технологии / Д. И. Козлов, Г. П. Аншаков, Я. А. Мостовой, А. В. Соллогуб. – М.: Машиностроение, 1998, С. 368.
4. Василенко И. В., Макаров В. А. Модели оценки надёжности программного обеспечения // Вестник новгородского государственного университета №28, 2004, С. 126-132.
5. Благодатских В. А. др. Стандартизация разработки программных средств: Учеб. пособие / В. А. Благодатских, В. А. Волнин, К. Ф. Поскакалов; – М.: Финансы и статистика, 2005. – С. 288.
6. М. Р. Мидлтон. Анализ статистических данных с использованием Microsoft Excel для Office XP. М.: Бином. Лаборатория знаний, 2005. С. 246-260.
7. Мостовой Я. А. Управление сложными техническими системами: конструирование программного обеспечения спутников ДЗЗ. Москва: – Техносфера, 2016, С. 319-323.
8. Мостовой Я. А., Бурзуева Н. Н. Определение надежности системы электронного документооборота Directum по результатам эксплуатации // Инфокоммуникационные технологии №3, 2016, С. 279-289. ForexAW.com // ForexAW.com – графики форекс, статьи и новости с рынков, ответы на вопросы и финансово-экономический справочник. URL: http://forexaw.com/TERMs/Industry/Technology/l1251_Android_%D0%90%D0%BD%D0%B4%D1%80%D0%BE%D0%B8%D0%B4_%D1%8D%D1%82%D0%BE (Дата обращения 12.03.2017).
9. Meet Android Studio // Android Studio URL: https://developer.android.com/studio/intro/index.html (Дата обращения 12.03.2017).
10. Android Studio The Official IDE for Android // Android Studio URL: https://developer.android.com/studio/index.html (Дата обращения 12.03.2017).
11. Android Open Source Project – Issue Tracker // Android Open Source Project – Issue Tracker URL: https://code.google.com/p/android/ (Дата обращения 12.03.2017).

Burzueva Natalia Novruzovna
Povolzhskiy state university of telecommunications and informatics, Russia, Samara
E-mail: nburzueva@gmail.com

Mostovoy Yakov Anatol'yevich
Povolzhskiy state university of telecommunications and informatics, Russia, Samara
E-mail: jakob.mostovoi@yandex.ru

Analysis of reliability for Android Studio development environment

Abstract. In this article two mutually explanatory of each other models of carrying-out errors (changes) in the software are considered: discrete and continuous ones, based on the parameters that follow from the Jelinsky-Morandi hypothesis, determined according to the statistics of the software error carrying-out in time. An expression is obtained that allows you to determine the probability of no errors in the software in a given time. The obtained expression allows to determine the probability of absence of errors in software after a specified time. It is proposed to assume that any formally registered correction of the code made by the developer is the elimination of an error or vulnerability, presence of which is also currently a error (change), either programmatic or algorithmic ones, which are sometimes "masked" as improvements. Under this approach, it is made possible to assess the reliability of the software in case when there is a statistics of its changes. The issues arising in calculating reliability indices of software are discussed in the article with regard to the Android Studio development environment. For performing calculations we collected statistics on the occurrence of errors for seven months with a critical and high priority for the Android Studio integrated development environment based on the results of the Android Open Source Project - Issue Tracker technical support site and with a probability of 0.9 identified a time interval, through which the errors (changes) would be absent in the software.

Keywords: software reliability; discrete and continuous models; Markov process; Jelinsky-Morandi model; least-square method; Android Studio; technical support