КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2018 Т. 10 № 3 С. 325–334
DOI: 10.20537/2076-7633-2018-10-3-325-334

Ки&М

СПЕЦИАЛЬНЫЙ ВЫПУСК

УДК: 621.3, 004.052, 519.718

Сравнение мобильных операционных систем на основе моделей роста надежности программного обеспечения

Д. Суччи, В. В. Иванов[a]

Университет Иннополис,
Россия, 420500, г. Иннополис, ул. Университетская, д. 1

E-mail: [a] v.ivanov@innopolis.ru

Получено 28.02.2018, после доработки — 24.05.2018.
Принято к публикации 24.05.2018.

Оценка надежности программного обеспечения (ПО) — важная составляющая процесса разработки современного программного обеспечения. Многие исследования направлены на улучшение моделей для измерения и прогнозирования надежности программных продуктов. Однако мало внимания уделяется подходам к сопоставлению существующих систем с точки зрения надежности ПО. Несмотря на огромное значение для практики (и для управления разработкой ПО), полной и проверенной методологии сравнения не существует. В этой статье мы предлагаем методологию сравнения надежности программного обеспечения, в которой широко применяются модели роста надежности программного обеспечения. Методология была оценена на примере трех мобильных операционных систем с открытым исходным кодом: Sailfish, Tizen, CyanogenMod.

Побочным продуктом исследования является сравнение трех мобильных операционных систем с открытым исходным кодом. Целью данного исследования является определение того, какая ОС является более надежной. Для этого были определены 3 вопроса и 8 метрик. С учетом сравнения этих метрик оказалось, что Sailfish в большинстве случаев является самой эффективной операционной системой. Напротив, Tizen показывает лучшее в 3 случаях из 8, но оказывается хуже других систем только в одном случае из 8.

Ключевые слова: надежность ПО, мобильные операционные системы

Работа выполнена при финансовой поддержке Университета Иннополис.

COMPUTER RESEARCH AND MODELING
2018 VOL. 10 NO. 3 P. 325–334
DOI: 10.20537/2076-7633-2018-10-3-325-334

Ки&М

SPECIAL ISSUE

UDC: 621.3, 004.052, 519.718

Comparison of mobile operating systems based on models of growth reliability of the software

G. Succi, V. V. Ivanov[a]

Innopolis University,
1 Universitetskaya st., Innopolis, 420500, Russia

E-mail: [a] v.ivanov@innopolis.ru

Received 28.02.2018, after completion — 24.05.2018.
Accepted for publication 24.05.2018.

Evaluation of software reliability is an important part of the process of developing modern software. Many studies are aimed at improving models for measuring and predicting the reliability of software products. However, little attention is paid to approaches to comparing existing systems in terms of software reliability. Despite the enormous importance for practice (and for managing software development), a complete and proven comparison methodology does not exist. In this article, we propose a software reliability comparison methodology in which software reliability growth models are widely used. The proposed methodology has the following features: it provides certain level of flexibility and abstraction while keeping objectivity, i.e. providing measurable comparison criteria. Also, given the comparison methodology with a set of SRGMs and evaluation criteria it becomes much easier to disseminate information about reliability of wide range of software systems. The methodology was evaluated on the example of three mobile operating systems with open source: Sailfish, Tizen, CyanogenMod.

A byproduct of our study is a comparison of the three analyzed Open Source mobile operating systems. The goal of this research is to determine which OS is stronger in terms of reliability. To this end we have performed a GQM analysis and we have identified 3 questions and 8 metrics. Considering the comparison of metrics, it appears that Sailfish is in most case the best performing OS. However, it is also the OS that performs the worst in most cases. On the contrary, Tizen scores the best in 3 cases out of 8, but the worst only in one case out of 8.

Keywords: reliability of software products, mobile operating systems

Citation: *Computer Research and Modeling*, 2018, vol. 10, no. 3, pp. 325–334 (Russian).

The work was supported by Innopolis University.

Введение

Сравнение надежности программных систем имеет первостепенное значение для современной программной инженерии. В настоящее время проблемы обеспечения качества программного обеспечения и прогнозирования поведения программных систем имеют большое значение, поскольку программное обеспечение включено в большинство областей человеческой деятельности, особенно в критически важных для безопасности областях (гидротехническая область, транспорт, химическая промышленность, системы предупреждения и т. д.).

Как определено в [Lyu, 1996], надежность программного обеспечения относится к вероятности безотказной работы программного обеспечения в течение определенного периода времени в заданных условиях. Надежность программного обеспечения обычно измеряется как количество дефектов, существующих в исходном коде программного продукта, или сбоев, которые происходят во время работы продукта [Wood, 1996]. Ранее проводились исследования по конкретным аспектам моделирования надежности программного обеспечения. В большинстве работ основное внимание уделяется оценке и сопоставлению моделей надежности программного обеспечения и меньше внимания уделяется сравнению программных продуктов с использованием разработанных моделей. В настоящее время нет обоснованной и проверенной методологии эффективного сопоставления надежности программных систем.

Цель данного исследования — формализация и обоснование методологии для сравнения надежности программных систем. Предлагаемый подход основан на анализе отчетов об ошибках, присутствующих в типичных системах отслеживания ошибок, которые в настоящее время используются в большинстве программных проектов. Валидация подхода проводилась на трех промышленных мобильных операционных системах:

- Cyanogen, основанная на Android операционная система (без необходимости предоставления услуг Google и с небольшими изменениями в ядре ОС);
- Sailfish, операционная система с открытым исходным кодом, разрабатываемая финской компанией Jolla; расширение проекта Nokia Maemo;
- Tizen, мобильная операционная система с открытым исходным кодом, разработанная компанией Samsung.

Эти три проекта находятся в стадии активного развития и имеют существенный размер (более 50 млн строк кода). Это дает возможность оценить надежность программного обеспечения на основе измеримых показателей роста надежности системы в ходе ее разработки.

Данная статья организована следующим образом. В начале изложены существующие предложения в сравнении надежности программных систем. Затем рассмотрена методология оценки надежности программных систем, которая основана на моделях роста надежности программного обеспечения. Описаны наборы данных, которые были использованы для оценки. Приведены результаты эмпирической оценки.

Обзор подходов к оценке надежности программных продуктов

Сравнение надежности программных систем проводится достаточно давно. Данная проблематика за последние 70 лет проникла практически во все области промышленной программной инженерии. Были разработаны подходы, охватывающие разные уровни жизненного цикла ПО. Методы охватывают различные аспекты программных систем: от низких уровней, сетевых и межсетевых аспектов, до уровня интеграции систем, вплоть до пользовательских интерфейсов.

Тем не менее работы по сравнению надежности программных систем, которые были применены в индустрии или по крайней мере в лабораторных условиях, с реальными проектами, немногочисленны. В большинстве таких исследований используются модели роста надежности программного обеспечения (далее Software Reliability Growth Models, SRGM). SRGM измеряют надежность программных продуктов; с их помощью можно рассчитать и сравнить коэффици-

енты моделей надежности для различных продуктов. Однако авторы большинства исследований обычно анализируют сами модели и редко фокусируются на сравнении программных систем. Таким образом, существует пробел в исследованиях, связанных с методологией сравнения программных систем в отношении их надежности. В этом разделе мы рассмотрим существующие подходы к оценке надежности программных систем, а также модели роста надежности программного обеспечения.

Общей целью сравнительного исследования надежности является установление общего подхода или системы критериев для получения измеримых и воспроизводимых результатов. Большинство SRGM — это параметрические модели с несколькими параметрами, которые могут быть интерпретированы и сопоставлены друг с другом для сравнения надежности моделируемых программных систем. Поэтому существующие методологии пытаются найти и проанализировать закономерности роста надежности программного обеспечения и использовать эти шаблоны в сравнении.

В работе [Zhou, Davis, 2005] выполнялась оценка программного обеспечения с открытым исходным кодом. Авторы использовали модель Вейбулла, предназначенную для отслеживания данных об ошибках. Были собраны данные о восьми проектах с открытым исходным кодом. Это исследование предназначалось для разработки и анализа моделей, связанных со временем обнаружения дефектов в ПО.

Основная идея подхода — оценить параметр формы модели Вейбулла в наборе данных и сравнить значения этого параметра у разных систем. Интересным результатом является сравнение проектов с открытым исходным кодом и проектов с закрытым исходным кодом. Проекты с открытым исходным кодом и проекты с закрытым кодом показали схожую форму кривой роста надежности. Более того, большинство проектов программного обеспечения с открытым исходным кодом стабилизируются на очень низком уровне скорости поступления ошибок.

Аналогичное исследование было представлено в работе [Syed-Mohamad, McBride, 2008]. Авторы используют две SRGM (вогнутую модель и S-образную модель) для сравнения двух продуктов с открытым исходным кодом. Основной вопрос исследования: имеет ли программное обеспечение с открытым исходным кодом существенно отличные коэффициенты скорости обнаружения дефектов по сравнению с проприетарным ПО? Авторы используют аналогичный подход, то есть подгонку (fitting) моделей к данным и сравнение параметров, связанных со скоростью обнаружения дефектов.

Однако они получили результаты, отличающиеся от предыдущего исследования. Продукты с открытым исходным кодом показывают различную скорость появления дефектов и, таким образом, демонстрируют наличие различий в процессах разработки. Авторы объясняют разницу в эффекте скорости обнаружения дефектов свойствами процесса разработки программного обеспечения с открытым исходным кодом. Разработчики программного обеспечения с открытым исходным кодом имеют тенденцию существенно (и крайне оперативно) менять исходный код между последующими релизами, чтобы удовлетворить новые ожидания пользователей; они часто могут переключиться на новые технологии. Другим фактором является большое сообщество разработчиков, которое легко выполняет изменения в исходном коде, но редко делает строгую проверку качества ПО. Это противоположно результатам, представленным в работе [Zhou, Davis, 2005]. Это является очевидным доказательством отсутствия всесторонних исследований в области методологий сравнения надежности программного обеспечения.

Одно из последних сравнительных исследований представлено в работе [Rossi et al., 2010]. Авторы стремились представить подход к исследованию надежности проектов с открытым исходным кодом с использованием нескольких SRGM. Их анализ направлен на определение моделей возникновения сбоев в трех продуктах OSS для прогнозирования поведения будущих релизов.

Они использовали данные из трех больших открытых датасетов (OpenSuse, Mozilla Firefox и OpenOffice.org) и применяли восемь моделей оценки надежности (Goel Okumoto, Goel Okumoto S-shaped, Gompertz, Hossain Dahiya, Logistic, Weibull S-shaped, Weibull more-S-shaped и модель Ямады).

Многие из существующих публикаций приводят значения параметров моделей. Однако сопоставить надежность различных продуктов бывает затруднительно, просто сравнивая значения установленных параметров, поэтому мы предлагаем более полную модель.

В работе [Ullah et al., 2012] исследуются критерий качества модели (goodness-of-fit, GoF) и возможности прогнозирования для восьми SRGM. Сравнение включает в себя пятьдесят различных наборов данных сбоев. Эти наборы данных содержат данные о дефектах, собранные на этапе тестирования системы из систем отслеживания ошибок проектов с открытым исходным кодом.

Данные о сбоях были смоделированы восемью SRGM. Авторы выбрали эти модели из-за распространенности среди многих моделей надежности программного обеспечения. Несмотря на то что исследование сосредоточено на сравнении моделей, оно служит хорошей демонстрацией широкого применения SRGM для многих проектов с открытым исходным кодом. Экспериментальные результаты анализируются и сравниваются с другими существующими моделями, чтобы показать, что предлагаемая модель дает лучшие прогнозы. Авторы также приводят значения параметров SRGM, что позволяет сравнивать надежность программных продуктов с использованием параметров моделей. Несмотря на то что большинство подходов используют модели повышения надежности программного обеспечения, они в основном сосредоточены на анализе и сравнении моделей, а не на программных продуктах.

Модели роста надежности ПО

Надежность программных систем может быть описана с использованием множества подходов, в том числе нечетких моделей, гранулированных моделей, регрессионного анализа, марковских моделей, нейронных сетей, и применяется в разнообразных контекстах.

Одной из самых сложных и важных задач является предсказание количества ошибок в разработанном или выпущенном программном продукте. SRGM особенно полезны в следующих аспектах, как указано в [Yamada, 2014]:

- определение оптимального времени для релиза очередной версии программного обеспечения для прогнозирования количества оставшихся дефектов;
- статистический контроль за ходом тестирования;
- оптимальное распределение трудозатрат на тестирование.

За последние 30 лет было разработано достаточно много моделей для анализа процесса обнаружения дефектов. Полный анализ SRGM можно найти в обзорах, подобных [Rossi et al., 2010], и в сравнительных исследованиях [Succi et al., 2003]. Тем не менее до сих пор не существует «единственно верного подхода» к построению модели надежности, которую можно было бы использовать для оптимизации критериев, важных на практике, таких как GoF, предсказание остаточных дефектов, устойчивость модели и др.

Процесс подсчета обнаруженных дефектов обычно представляется неоднородным пуассоновским процессом (NHPP). Область SRGM, основанная на NHPP, привлекает большой интерес исследователей. Предложены десятки основанных на NHPP моделей SRGM для оценки и прогнозирования скорости появления дефектов. Эти модели были исследованы и сопоставлены друг с другом в многочисленных исследованиях, таких как [Succi et al., 2003]. Широко распространенное утверждение в исследованиях SRGM заключается в том, что лучшие модели, как правило, хорошо работают при определенном наборе предположений, и нет модели, которая была бы лучшей в любой ситуации.

Измерение и прогнозирование роста надежности программных продуктов на основе SRGM

Как уже упоминалось выше, цель исследования — сравнить три программных продукта с открытым исходным кодом с точки зрения надежности с использованием нескольких моделей

роста надежности (SRGM), принимая во внимание, что каждая модель отличается от других по качеству. Модель роста надежности программного обеспечения представляет собой математическую модель, которая описывает события появления сбоев ПО или обнаружения дефектов на этапах тестирования и отладки системы. Такие модели предназначены для оценки роста надежности в ходе тестирования, а также для прогнозирования надежности программного обеспечения на более поздних этапах разработки продукта. SRGM можно разделить на две группы: модели с непрерывным временем между отказами (измеренными как время выполнения теста или календарного времени) и модели с дискретным временем (измеренные в нескольких сериях тестов).

В SRGM с дискретным временем суммарное количество дефектов, поступающих от момента времени 0 до момента t, обозначается через $n(t)$ и обычно моделируется как результат случайного процесса. Данный процесс обычно считается неоднородным пуассоновским процессом (NHPP); в этом случае SRGM представляется параметрической функцией среднего значения $m(t)$. Функция среднего значения обозначает ожидаемое количество дефектов, обнаруженных до момента времени t. Форма кривой зависит от свободных параметров, которые оцениваются из набора данных с помощью методов машинного обучения.

Мы кратко рассмотрим модели, основанные на NHPP, из-за их более высокой актуальности для нашего исследования. Этот класс моделей можно разделить на два подкласса: S-образные модели и вогнутые модели. Одним из первых вкладов была модель Гоэля–Окумото, где функция среднего значения имела следующий вид:

$$m(t) = a\ (1 - e^{-bt}).$$

Этот вид функции определяется из следующего уравнения:

$$\frac{d}{dt} m(t) = b\big(a - m(t)\big).$$

Вогнутые модели описывают ситуацию, когда изначально дефекты обнаруживаются быстро, но по мере их обнаружения вероятность найти дефект уменьшается, поэтому скорость обнаружения падает. S-образные модели являются сначала выпуклыми, а затем вогнутыми, то есть скорость обнаружения дефектов сначала увеличивается, что отражает начальную фазу обучения, а затем падает. Вначале тестировщик не знаком с программным обеспечением, поэтому использование системы ограничено, обнаружение происходит медленно, но через некоторое время он получает больше опыта и знаний о поведении ПО. Это подразумевает более высокую скорость обнаружения дефектов до тех пор, пока не будут обнаружены почти все дефекты. Затем процесс замедляется, становится все труднее и труднее обнаруживать новые дефекты. Таким образом, частота обнаружения сначала увеличивается, а затем уменьшается. Разумеется, вторая производная функции $m(t)$ отвечает за ее форму. Некоторые из рассматриваемых нами функций могут быть вогнутыми или S-образными в зависимости от значений их параметров.

Параметры SRGM были предметом изучения в ряде сравнительных исследований. В работе [Jatain, Mehta, 2014] оценивались параметры, которые влияют на надежность программного обеспечения. В сравнительном исследовании [Ullah et al., 2012] были выбраны восемь моделей: Musa Okumoto, Inflection S-Shaped, Goel Okumoto, Delayed S-Shaped, Logistic, Gompertz, экспоненциальная модель и обобщенная модель Goel.

В таблице 1 представлены модели, которые применялись в данном исследовании, и их параметры. Список выбранных моделей был разработан на основе работ [Succi et al., 2003; Wood, 1996]. Важным критерием для оценки качества модели, который используется в многочисленных исследованиях SRGM, является расхождение между данными и моделью (Goodness of Fit, GoF). Однако в случае анализа надежности программных продуктов важны и другие критерии, которые будут рассмотрены ниже.

Таблица 1. SRGM-модели, используемые в исследовании

Модель	Формула
Goel–Okumoto (G-O)	$a(1-e^{-bt})$
G-O S-образная (GO-S)	$a(1-(1+bt)e^{-bt})$
Логистическая (L)	$\frac{a}{1+be^{-ct}}$
Hossain–Dahiya (HD)	$a\frac{1-e^{-bt}}{1+ce^{-ct}}$
Вейбулла (W)	$a(1-e^{-bt^c})$
S-образная Вейбулла (W-S)	$a(1-(1+bt^c)e^{-bt^c})$
Экспоненциальная модель Ямада (YE)	$a\left(1-e^{-b\left(1-e^{-ct}\right)}\right)$
Модель Ямады–Релея (YR)	$a\left(1-e^{-b\left(1-e^{-c\frac{t^2}{2}}\right)}\right)$

Критерии оценки и результаты экспериментов

Рассмотрим критерии оценки качества SRGM. Из критериев, предложенных [Succi et al., 2003], мы выбрали следующие:

- соответствие данных и модели (GoF);
- относительная точность (RPoF);
- покрытие данных моделью (CoF).

Соответствие данных и модели (GoF) показывает, насколько модель соответствует исходным данным. Единицей, используемой для GoF, является количество ошибок. Традиционно GoF рассчитывается как сумма квадратов остатков, деленная на количество степеней свободы.

Относительная точность (RPoF) — это область 95%-го доверительного интервала (95 % полосы, моделируемой в течение анализируемого времени). Единица, используемая для этой меры, основана на произведении общего времени (выраженного в днях) и количества дефектов. Поэтому модели с низким значением RPoF обычно обладают высокой способностью предоставлять полезную информацию о возникновении дефектов.

Покрытие данных моделью (CoF) — это степень, в которой 95%-й доверительный интервал модели покрывает обнаруженные дефекты (данные). Единицей, используемой для покрытия, является доля от общего числа дефектов в пределах доверительного интервала. RPoF и CoF представляют собой два взаимодополняющих аспекта модели, которые необходимо рассматривать совместно. Действительно, очень большая полоса могла покрывать практически 100 % от точек данных, но при этом она имела бы слишком высокую относительную точность. И наоборот, низкие значения RPoF обычно приводят лишь к частичному покрытию данных (CoF < 100 %).

Важно подчеркнуть, что эти критерии охватывают различные аспекты моделей, которые в общем случае могут противоречить друг другу. Например, очень большое значение охвата (CoF) может быть получено с помощью очень большего доверительного интервала (RPoF). Однако такая ситуация может «штрафоваться» за счет низкого значения соответствия модели дан-

ным (GoF). Поэтому мы считаем, что в целом эти критерии представляют собой четкое описание качества SRGM. Как уже упоминалось, мы проверяем методологию на трех мобильных операционных системах с открытым исходным кодом:

– CyanogenMod,
– Sailfish,
– Tizen.

Чтобы вычислить метрики о размере проектов, мы проанализировали репозитории исходного кода. Проекты написаны на множестве языков программирования (включая C, C ++ и Java). Для каждого проекта приводятся количество файлов и число строк кода (таблица 2).

Ориентируясь на критерии качества SRGM, в таблицу 3 мы включили характеристики надежности трех операционных систем.

Таблица 2. Размеры наборов данных

Проект	Число файлов с исходным кодом	Число строк исходного кода	Общее число дефектов
Sailfish	864998	97041969	3425
Tizen	987273	122166906	10092
CyanogenMod	457649	50523609	19771

Таблица 3. Качество SRGM-моделей

Модель	Sailfish			Tizen			CyanogenMod		
	GoF	RPoF	CoF	GoF	RPoF	CoF	GoF	RPoF	CoF
G-O	4.07	1047	1	46.12	154	0.99	48.94	1296	0.97
GO-S	6.08	939	1	36.81	131	0.99	40.66	1583	0.98
L	4.72	945	0.81	35.19	1252	0.56	37.88	776	0.41
HD	5.31	1363	1	34.99	348	1	38.16	2008	0.96
W	3.86	2393	1	46.27	7211	0.99	49.57	5475	0.88
W-S	3.53	8450	1	44.81	3567	1	60.08	5571	1
YE	3.63	10263	1	36.11	45079	1	50.34	57597	1
YR	5.00	16200	1	36.70	33926	1	42.92	131810	1

Таблица 4. Значения параметров SRGM-моделей

Модель	Параметр модели	Sailfish	Tizen	CyanogenMod
G-O*	b (positive)	0.05	0.003	0.0068
GO- S*	b (positive)	0.025	0.013	0.016
L*	c (positive) b (negative)	0.087 3.99	0.026 14.09	0.045 27.56
HD*	b (positive) c (negative)	0.034 0.77	0.021 9.49	0.018 3.66
W*	c (positive) b (positive secondarily influential)	0.71 0.06	0.76 0.007	0.67 0.0084
W-S*	c (positive) b (positive secondarily influential)	0.38 0.625	0.89 0.016	0.61 0.08
YE	c (positive) b (positive secondarily influential)	0.0519 0.59	0.004 0.47	0.0044 1.1
YR*	c (positive) b (positive secondarily influential)	0.0015 1.16	0.0001 0.91	0.0001 0.87

Из анализа результатов (таблица 3) представляется, что в рассматриваемых случаях наилучшим образом подходят модели GO-S, L, HD и W-S. Теперь мы можем рассмотреть результаты моделей в терминах параметров (таблица 4). Параметр *b*, представленный в этой таблице, характеризует скорость, с которой обнаруживаются дефекты в ПО в ходе тестирования. Можно заметить, что для Sailfish этот параметр выше, чем для других ОС. Это выполняется, в частности, для моделей, отмеченных звездочкой (*) в таблице 4, а именно GO-S, L, HD и W-S. Для модели L параметр *b* оказывает противоположное влияние на скорость выявления ошибок. Таким образом, с точки зрения способности обнаруживать ошибки проект ОС Sailfish оказался лучше. Что касается худшей ОС, то здесь трудно различить Tizen и CyanogenMod. С одной стороны, обе системы оказались лучше в одном случае, но Tizen чаще всего оказывается хуже. С другой стороны, Tizen лучше всего работает для одной из самых точных моделей (W-S), при этом, когда Tizen оказывается хуже, оказывается достаточно близко к CyanogenMod. В целом можно заключить, что CyanogenMod — наихудшая операционная система с точки зрения скорости обнаружения сбоев.

Заключение

Основная цель исследования — разработка и валидация новой методологии для сравнения надежности программных продуктов. В этом исследовании представлена и проверена методология сравнения программных систем с точки зрения их надежности. Предлагаемая методология имеет следующие функции, которые соответствуют первоначальной цели исследования: она обеспечивает определенный уровень гибкости и абстракции, сохраняя при этом объективность, то есть обеспечивая измеримые критерии сравнения. Наконец, сопоставляя методологию сравнения (как именно проводить сравнение надежности) с набором SRGM и критериями оценки (что является результатом сравнения), становится намного проще распространять информацию о надежности широкого спектра программных систем. Во время применения методологии к мобильным ОС мы обнаружили, что:

- Tizen OS — самая эффективная мобильная операционная система из трех рассмотренных с точки зрения надежности основных модулей и компонентов;
- CyanogenMod — наихудшая операционная система с точки зрения скорости обнаружения сбоев;
- с точки зрения способности обнаруживать ошибки проект Sailfish оказался лучшим.

Будущие работы будут сосредоточены на моделировании данных с несколькими релизами, что на данный момент является основным ограничением к применению полученных результатов на практике.

Список литературы (References)

Jatain A., Mehta Y. Metrics and models for software reliability: A systematic review // Issues and Challenges in Intelligent Computing Techniques (ICICT). — 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT). Feb 2014. — P. 210–214.

Lyu M. R. (*ed.*) Handbook of Software Reliability Engineering. — McGraw-Hill, Inc., Hightstown. — NJ, USA, 1996.

Rossi B., Russo B., Succi G. Modelling failures occurrences of open source software with reliability growth // Agerfalk P., Boldyreff C., Gonzalez-Barahona J. M., Madey G. R., Noll J. (eds.). — Open Source Software: New Horizons. — Vol. 319. — IFIP Advances in Information and Communication Technology. — P. 268–280. — Springer Berlin Heidelberg, 2010.

Succi G., Pedrycz W., Stefanovic M., Russo B. An investigation on the occurrence of service requests in commercial software applications. — Empirical Softw. Engg., 8 (2). — P. 197–215. — June 2003.

Syed-Mohamad S. M., McBride T. Reliability growth of open source software using defect analysis // Computer Science and Software Engineering. — 2008 International Conference on Computer Science and Software Engineering (CSSE 2008). — Vol. 2. — P. 662–667. — IEEE, 2008.

Ullah N., Morisio M., Vetro A. A comparative analysis of software reliability growth models using defects data of closed and open source software // Software Engineering Workshop (SEW). — 2012 35th Annual IEEE. — P. 187–192. — IEEE, 2012.

Wood A. Predicting software reliability. Computer. — 29 (11). — P. 69–77. — Nov. 1996.

Yamada S. Software Reliability Modeling Fundamentals and Applications. — Springer Japan, 2014.

Zhou Y., Davis J. Open source software reliability model: an empirical approach // ACM SIGSOFT Software Engineering Notes. — Vol. 30. — P. 1–6. — ACM, 2005.