УДК 004.42+004.9-027.45

МЕТОДИКА ОБЕСПЕЧЕНИЯ ЭКСПЛУАТАЦИОННОЙ НАДЁЖНОСТИ ПЛАНИРУЕМЫХ К РАЗРАБОТКЕ ПРИКЛАДНЫХ КОМПЬЮТЕРНЫХ ПРОГРАММ ДЛЯ ИНФОРМАЦИОННЫХ СИСТЕМ

С.М. Боровиков
доцент кафедры ПИКС БГУИР, кандидат технических наук

В.О. Казючиц
аспирант кафедры ПИКС БГУИР, магистр технических наук

С.К. Дик
доцент кафедры ЭТТ БГУИР, кандидат физико-математических наук, доцент

С.С. Дик
проект-менеджер компании «Itransition», магистр техники и технологии

Белорусский государственный университет информатики и радиоэлектроники, Республика Беларусь
Компания «Itransition», Республика Беларусь
E-mail: bsm@bsuir.by

С.М. Боровиков

Доцент кафедры проектирования информационно-компьютерных систем БГУИР. Основная область научных интересов – прикладные математические методы в проектировании изделий радиоэлектроники, включая алгоритмы статистического прогнозирования надёжности изделий электронной техники и оценку надёжности прикладного программного обеспечения на ранних этапах его разработки. Руководитель разработки программных комплексов по автоматизированному расчёту и обеспечению надёжности электронных устройств: система АРИОН (2008-2009 гг.), система АРИОН-плюс (2011-2015 гг.).

В.О. Казючиц

Окончил БГУИР (2017 г.), в настоящее время является аспирантом этого университета, магистр технических наук. Проводит научные исследования по прогнозированию надёжности полупроводниковых приборов большой мощности и разработке прикладных компьютерных программ для автоматизированной оценки их надёжности.

С. К. Дик

Окончил Минский радиотехнический институт по специальности «Радиотехника», руководит научными исследованиями в области лазерной медицины и биомедицинской оптики.

С.С. Дик

Окончил Белорусский государственный университет информатики и радиоэлектроники (2016 г.), и аспирантуру при этом университете (2020 г.), магистр техники и технологии. Работает в компании «Itransition» в должности проект-менеджера, занимается разработкой и внедрением программного обеспечения в различные сферы деятельности людей.

Аннотация. Прикладные компьютерные программы, используемые для информационных систем, в том числе для технологий Big Data, содержат десятки тысяч–миллионы строк программного кода и поэтому после написания в них всегда имеются скрытые ошибки, которые не могут быть определены компилятором языка программирования. Задача этапа тестирования состоит в том, чтобы выявить и устранить наиболее критичные ошибки с точки зрения выполнения компьютерной программой своих функций и уменьшить число оставшихся в программе скрытых ошибок. Скрытые ошибки, оставшиеся в программе после её тестирования,

иногда могут себя проявлять в зависимости от степени изменчивости исходных данных и нагрузки, которую оказывает эксплуатационная среда на компьютерную программу (запись информации на электронные носители, печать на принтере, ожидание в очереди и т.д.). Наличие этих скрытых ошибок определяет уровень эксплуатационной надёжности компьютерных программ. Разработчики информационных систем, в том числе аналитических систем, используемых в технологиях Big data, хотели бы знать, какое время и, следовательно, финансовые затраты потребуются на тестирование планируемых к разработке прикладных компьютерных программ с целью обеспечения заданного уровня их эксплуатационной надёжности. В работе поясняется предлагаемая методика, используя которую можно получить ответ на эти вопросы. Методика разработана на основе экспериментальных данных о надёжности прикладных компьютерных программ разного функционального назначения.

Ключевые слова: прикладные компьютерные программы, начальная надёжность, обеспечение эксплуатационной надёжности, эффективность тестирования, прогнозирование времени тестирования.

Работа проводилась в рамках договора № Ф20МВ-021 на выполнение НИР «Статистические модели надёжности прикладных программных средств и их использование для оценки ожидаемой безотказности компьютерных программ на ранних этапах их разработки» в соответствии с решением научного совета БРФФИ по результатам конкурса «БРФФИ–Минобразование М-2020 (протокол № 1 от 22.04.2020).

Введение.

Аналитические системы, используемые для технологий больших данных (Big Data), относятся к классу сложных информационно-компьютерных систем, в которых вклад программного обеспечения в ненадёжность систем может составлять 40 и более процентов [1]. Для принятия решения о целесообразности разработки и уровне эффективности функционирования информационных систем в заданных условиях необходимо на ранних этапах её проектирования оценить ожидаемую эксплуатационную надёжность планируемых к разработке прикладных компьютерных программ. Под эксплуатационной надёжностью будем понимать тот уровень надёжности, который прикладная компьютерная программа покажет на начальном этапе её эксплуатации, то есть после выполнения этапа тестирования и отладки.

Актуальность.

Прикладные компьютерные программы, используемые для информационных систем, в том числе для технологий Big Data, содержат десятки тысяч–миллионы строк программного кода и поэтому после написания программ и устранения синтаксических ошибок в них всегда имеются скрытые ошибки. На этом этапе компьютерная программа характеризуется начальным уровнем надёжности, который как правило не отвечает требованиям заказчика. Поэтому необходимо тестирование с целью поиска скрытых смысловых (арифметических, логических и др.) ошибок. Задача этого этапа состоит в том, чтобы выявить и устранить наиболее критичные ошибки с точки зрения правильности выполнения программой своих функций, уменьшить число скрытых ошибок, оставшихся в компьютерной программе, и в итоге – обеспечить приемлемый уровень эксплуатационной надёжности прикладной компьютерной программы.

Проектировщики аналитических систем и программного обеспечения к ним хотели бы знать ожидаемый уровень надёжности прикладной компьютерной программы ещё до написания кода программы и выполнения её тестирования. Разработчиков аналитических систем интересуют также затраты времени и, следовательно, затраты средств, которые необходимы для проведения тестирования с целью достижения заданного уровня эксплуатационной надёжности прикладной компьютерной программы. Поэтому наличие методики, которая даст ответ на вопрос о требуемом времени тестирования планируемой к разработке прикладной компьютерной программы с целью обеспечения её эксплуатационной надёжности, является актуальной задачей.

Подходы к разработке методики обеспечения эксплуатационной надёжности прикладных компьютерных программ.

Методика разработана на основе ранее проведённых исследований [2–13 и др.] по оценке ожидаемой надёжности и тестированию планируемых к разработке прикладных компьютерных программ. Результаты исследований в виде модели прогнозирования надёжности прикладной компьютерной программы [2, 5] и модели прогнозирования времени тестирования программы с целью обеспечения заданного уровня её эксплуатационной надёжности [3] были получены с использованием экспериментальных данных о надёжности и тестировании прикладных компьютерных программ разных областей применения [14]. Модель надёжности прикладной компьютерной программы, приводимая в [2, 5], учитывает:

- особенность организации, которая будет разрабатывать программу;
- основные характеристики программы: сложность, средства разработки, новизну, использование стандартных модулей;
- быстродействие процессора, используемого компьютером;
- отрасль применения компьютерной программы, что определяет изменчивость реального потока наборов исходных данных на входе программы и рабочую нагрузку на компьютерную программу со стороны эксплуатационной среды (ввод-вывод данных и нахождение этих операций в очереди, наличие состояний ожиданий, загрузка-выгрузка программы и/или её модулей из памяти и т.д.).

Модели, приводимые в [3], позволяют примерно оценить процессорное и календарное время тестирования, необходимое для получения требуемого коэффициента эффективности тестирования *Q*, гипотетически обеспечивающего заданный уровень эксплуатационной надёжности компьютерной программы ($\lambda_{экс}$). Согласно [2], коэффициент эффективности тестирования *Q*, показывает, во сколько раз уменьшается интенсивность отказов компьютерной программы после выполнения её тестирования ($\lambda_{экс}$) относительно начальной интенсивности отказов λ_0 (до начала тестирования).

Основные структурные части методики.

Предлагаемая методика включает следующие структурные части:

- область применения;
- нормативные ссылки;
- термины, определения и сокращения;
- общие положения;
- количественные показатели описания надёжности прикладных компьютерных программ;
- подготовка исходных данных, используемых для оценки надёжности планируемой к разработке компьютерной программы;
- прогнозирование начальной плотности ошибок прикладной компьютерной программы;
- определение начальной интенсивности отказов прикладной компьютерной программы;
- прогнозирование процессорного времени тестирования для обеспечения заданной эксплуатационной надёжности прикладной компьютерной программы;
- определение календарного времени тестирования;
- пример применения методики;
- приложения.

Ниже приводится пояснение некоторых структурных частей методики.

Количественные показатели, используемые для описания надёжности прикладных компьютерных программ.

Согласно [15] для описания ожидаемой надёжности планируемой к разработке прикладной компьютерной программы (далее, компьютерной программы) будем использовать интенсивность проявления оставшихся в программе скрытых ошибок (λ),

а также вероятность отсутствия факта проявления ошибок программы в течение заданного календарного времени τ в предположении, что известно среднее число сеансов (прогонов) компьютерной программы за один час – η (размерность: 1/ч).

После написания кода компьютерной программы и устранения ошибок, вызываемых нарушением правил языка программирования, компьютерная программа характеризуется начальной интенсивностью отказов – λ_0. Последующее тестирование компьютерной программы позволяет устранить наиболее критичные ошибки с точки зрения правильности выполнения программой своих функций. В компьютерных программах большого объёма после тестирования всегда остаётся какая-то доля скрытых ошибок, которые могут себя иногда проявлять в зависимости от набора исходных данных и нагрузки на компьютерную программу со стороны эксплуатационной среды. Компьютерная программа после тестирования характеризуется эксплуатационной интенсивностью проявления ошибок программы (интенсивностью отказов) – $\lambda_{экс}$.

В качестве основной характеристики безошибочности выполнения программой своих функций согласно [15] будем рассматривать вероятность того, что компьютерная программа правильно (безошибочно) выполнит обработку одного произвольного набора исходных данных из числа наборов, которые могут поступать в условиях функционирования компьютерной программы в составе информационной системы. Эту вероятность p_1 согласно [16] определяют по экспоненциальной функции:

$$p_1 = \exp\left(\lambda_{экс} \cdot t_1\right), \tag{1}$$

где t_1 – среднее время обработки компьютерной программой одного набора исходных данных.

Вероятность p_1 отражает число оставшихся в компьютерной программе скрытых ошибок и их размещение в структуре программы, изменчивость реального потока наборов исходных данных на входе компьютерной программы и нагрузку на компьютерную программу со стороны эксплуатационной среды (загрузка и выгрузка программы из памяти, нахождение операций ввода-вывода в очереди, наличие состояний ожидания и т.д.).

Уточнив интенсивность обращения к компьютерной программе, т. е. среднее число сеансов («прогонов») программы за один час η (1/ч) в процессе функционирования информационно-компьютерной системы, можно определить вероятность того, что оставшиеся дефекты в компьютерной программе не проявятся в течение заданного календарного времени τ:

$$P(\tau) = \left(p_1\right)^{\eta \cdot \tau}. \tag{2}$$

Если значение вероятности $P(\tau)$ не отвечает требованиям заказчика, то как выход из положения, следует предпринять меры по увеличению вероятности p_1 путём дополнительного тестирования компьютерной программы.

Подготовка исходных данных, необходимых для оценки надёжности планируемой к разработке компьютерной программы.

В качестве исходных данных на этом этапе должны быть:

1. *Область применения (назначения) компьютерной программы.* Она определяет выбор базовой (усреднённой) начальной плотности ошибок A и значения суммарного коэффициента увеличения интенсивности отказов (K_Σ), обусловленного совместным действием изменчивости входных данных и рабочей нагрузки на компьютерную программу (таблица 1).

2. Обобщённый алгоритм выполнения компьютерной программы своих функций, предполагаемый (прогнозный) объём программы в строках исполняемого кода, используемый язык программирования.

Наличие этой информации позволит получить прогнозные оценки следующих характеристик:

- объёма программы в строках исполняемого кода – *L* (англоязычный вариант Lines Of Code – LOC);

- количества команд (операторов) программы *B*, определяемых для экстремального набора исходных данных на входе программы.

Таблица 1. Выбор коэффициентов для прикладных компьютерных программ различных областей применения

Область применения компьютерной программы	Значение *A*, ошибка / KLOC	Коэффициент K_Σ	Средний процент времени «прогона» компьютерной программы при её тестировании в течение календарной продолжительности
1. Авиация	12,8	5,23	8
2. Мониторинг и обеспечение безопасности	9,2	1,00	43
3. Телекоммуникации, мобильные устройства	7,8	11,5	3,5
4. Управление производственными процессами	1,8	3,17	14
5. Автоматизированные системы управления	8,5	19,2	2,5
6. Разработка программ, моделирование, обучение	12,3	14,1	3
Среднее	12,8	8,83	12

Общий прогнозный объём компьютерной программы *L* в строках кода (LOC) определяется исходя из количества и объёма функций, реализуемых программой, и определяется по формуле [17]

$$L = \sum_{i=1}^{n} v_i \,, \tag{3}$$

где v_i – объём в строках кода (LOC) для выполнения компьютерной программой *i*-й функции.

Значения v_i могут быть взяты из каталога функций программного обеспечения, включённого в документ [17]. В каталоге приводятся обобщённые (усреднённые) значения v_i в зависимости от используемого языка программирования и кода функции. Например, для функции с кодом 102 «Контроль, предварительная обработка и ввод информации» при использовании языка Visual C++ (Microsoft) v_i = 550 LOC, а для функции с кодом 203 «Обработка наборов и записей базы данных» при использовании языка Java v_i = 2370 LOC. Конечное значение *L* может быть представлено также в тысячах строках кода (англоязычный вариант kilolines of code – KLOC).

Прогнозное количество команд (операторов) программы *B* рекомендуется определять, как произведение

$$B = L\, E_L\, E_{\text{ц}} \,, \tag{4}$$

где E_L – коэффициент расширения кода программы (увеличения числа команд компьютерной программы) относительно числа строк кода L, определяется используемым языком программирования; $E_{ц}$ – коэффициент увеличения числа выполняемых процессором команд за счёт наличия в компьютерной программе циклов, ветвлений и других особенностей (число циклов и их размер, объём тела циклов, число ветвлений, особенности интерфейса и т. д.).

Выбор E_L определяется языком написания программного кода, например для Си $E_L \geq 2{,}5$; для Fortran, Cobal $E_L \geq 3{,}0$; для Ada $E_L \geq 4{,}5$; для Си++ $E_L \geq 6{,}0$. В случае неопределённости рекомендуется принять $E_L = 10$ [2, 18].

Значение $E_{ц}$ выбирается на основе экспертной оценки с учётом алгоритма выполнения компьютерной программой своих функций, числа и размера циклов, объём кода тела циклов, наличие ветвлений, вида интерфейса и т.п.). В зависимости от особенностей реальных потоков набора исходных данных, поступающих в компьютерную программу, значение $E_{ц}$ может достигать десятков–сотен единиц.

*3. Характеристика производственной среды разработки программного обеспечения: особенность организации, разрабатывающей компьютерную программу; характеристика программистов и их квалификация***.**

Характеристику производственной среды рекомендуется учитывать с помощью метрики (коэффициента) среды разработки программы D, представляющей произведение двух других коэффициентов [2, 6, 14]:

$$D = K_{орг} \cdot K_{кв.прог}, \quad (5)$$

где $K_{орг}$ – коэффициент, характеризующий особенность организации, разрабатывающей компьютерную программу; $K_{кв.прог}$ – коэффициент, учитывающий квалификацию и опыт программистов.

Значения $K_{орг}$ могут быть выбраны по таблице 2, либо получены методом экспертных оценок из условия $0{,}5 \leq K_{орг} \leq 2$ [14].

Таблица 2. Выбор коэффициента $K_{орг}$ метрики производственной среды D.

Характеристика организации, разрабатывающей компьютерную программу	Значение $K_{орг}$
1. Группа программистов работает в организации, обеспечивает ее потребности в программном обеспечении и отвечает за программу	0,76
2. Группа программистов имеет опыт работы по разработке программ, но не связана с пользователем программы	1,00
3. Группа программистов обладает опытом работы с компьютером, но может быть не знакома с приложением, для которого разрабатывается программа. Программа должна использоваться во взаимосвязанном комплексе технических средств, программного обеспечения и действий оператора	1,30

Для выбора значения коэффициента $K_{кв.прог}$ следует воспользоваться таблицей 3 [2].

Таблица 3. Значения коэффициента $K_{\text{кв.прог}}$

Квалификация и опыт программиста	Значение $K_{\text{кв.прог}}$
1. Специалист, освоивший программирование на уровне программы учебной дисциплины высшего технического учебного заведения	2,0
2. Младший программист (*Junior Developer*)	1,3
3. Программист (*Middle Developer*)	1,0
4. Ведущий программист (*Senior Developer*)	0,7

4. Характеристика планируемой к разработке компьютерной программы.

Характеристику планируемой к разработке компьютерной программы рекомендуется описывать с помощью метрики *S*, представляющей произведение коэффициентов, учитывающих важнейшие факторы, влияющие на надёжность программы [2, 6, 14]:

$$S = K_{\text{слож}} \cdot K_{\text{С.Р}} \cdot K_{\text{нов}} \cdot K_{\text{мод}}, \quad (6)$$

Пояснение коэффициентов вида K_i, входящих в (6), и рекомендации по выбору их значений приведены в таблице 4.

Таблица 4. – Коэффициенты, используемые для определения метрики характеристик программы

Обозначение коэффициента	Пояснение	Номер таблицы для выбора значения	Примечание
$K_{\text{слож}}$	Учитывает степень сложности компьютерной программы	5, 6	Смотри формулу (6)
$K_{\text{С.Р}}$	Характеризует используемые средства разработки компьютерной программы	7	–
$K_{\text{нов}}$	Характеризует степень новизны компьютерной программы	8	–
$K_{\text{мод}}$	Учитывает степень использования стандартных модулей в компьютерной программе	9	–

Для выбора значений коэффициента $K_{\text{слож}}$ необходимо, пользуясь таблицей 5, вначале уточнить число характеристик сложности, присущих планируемой к разработке компьютерной программе. Значение $K_{\text{слож}}$ следует определять по формуле

$$K_{\text{слож}} = 1 + \sum_{i=1}^{m} k_i, \quad (7)$$

где k_i – коэффициент повышения сложности компьютерной программы за счёт наличия *i*-й характеристики (см. таблицу 6); *m* – число характеристик, присущих компьютерной программе.

Таблица 5. Описание характеристик сложности прикладных компьютерных программ

Номер характеристики	Описание характеристики компьютерной программы
1	Наличие сложного интеллектуального языкового интерфейса с пользователем
2	Обеспечение телекоммуникационной обработки данных и управление удалёнными объектами
3	Обеспечение существенного распараллеливания вычислений
4	Криптография и другие методы защиты информации
5	Моделирование объектов и процессов
6	Обеспечение настройки программного обеспечения на изменения структур входных и выходных данных
7	Обеспечение переносимости ПО
8	Реализация особо сложных инженерных и научных расчётов

Таблица 6. Коэффициенты повышения сложности компьютерной программы (k_i)

Характеристика повышения сложности компьютерной		Значения k_i
1. Функционирование компьютерной программы в расширенной операционной среде (связь с другими компьютерными программами)		0,08
2. Интерактивный доступ		0,06
3. Обеспечение хранения, ведения и поиска данных в сложных структурах		0,07
4. Наличие у компьютерной программы одновременно нескольких характеристик, указанных в таблице 5 (из пп. 4.1 – 4.3 выбирается одно из значений)	4.1. Две характеристики	0,12
	4.2. Три характеристики	0,18
	4.3. Свыше трёх характеристик	0,26

Выбор коэффициентов $K_{С.Р}$, $K_{нов}$ и $K_{мод}$ может быть сделан соответственно по таблицам 7, 8 и 9.

Таблица 7. Выбор значений коэффициента $K_{С.Р}$

Средства разработки компьютерной программы	Значение коэффициента $K_{С.Р}$ в зависимости от ОС		
	IBM-PC, Windows	Функционирование программы в сетях	
		локальных	глобальных
Языки высокого (Си++, Паскаль)	1,0	1,2	1,3
Языки 4GL (Visual Basic, Delphi)	0,8	0,95	1,1
Системы программирования на основе СУБД типа Foxpro	0,45	0,55	0,65
Системы программирования на основе СУБД типа Orac1e, SQLServer	0,4	0,5	0,6
Объектно-ориентированные технологии (COM/DCOM, CORBA)	0,55	0,6	0,7
Прочие CASE-средства	0,19	0,22	0,25

Таблица 8. Коэффициенты, учитывающие новизну компьютерной программы ($K_{нов}$)

Степень новизны	Использование компьютерной программы		Значение $K_{нов}$
	на основе нового типа ПК*	в среде новой ОС*	
Принципиально новые компьютерные программы, не имеющие подобных аналогов	+	+	1,58
	–	+	1,44
	+	–	1,10
	–	–	1,0
Компьютерные программы, являющиеся развитием определённого параметрического ряда компьютерных программ	+	+	1,0
	–	+	0,81
	+	–	0,72
Компьютерные программы, являющиеся развитием определённого параметрического ряда компьютерных программ, разработанных для ранее освоенных типов конфигурации ПК и ОС	–	–	0,63
* – Принятые сокращения: ПК – персональный компьютер, ОС – операционная система			

Таблица 9. Значение коэффициента $K_{мод}$, учитывающего степень использования стандартных модулей

Степень охвата стандартными модулями реализуемых функций разрабатываемой компьютерной программы	Значение $K_{мод}$
1. От 60% и выше	0,55
2. От 40 до 60%	0,65
3. От 20 до 40%	0,77
4. До 20%	0,9
5. Не используются стандартные модули для реализации функций разрабатываемой компьютерной программы	1

Прогнозирование начальной плотности ошибок.

Прогнозируемая начальная плотность ошибок F_0 определяется в соответствии с моделью RL-92-52 [14] как произведение метрик *A*, *D* и *S*:

$$F_0 = A \cdot D \cdot S, \text{ ошибка/LOC}, \quad (8)$$

где базовая плотность ошибок (метрика) *A* выбирается по таблице 1 в зависимости от области применения компьютерной программы, а метрики *D* и *S* определяются соответственно по формулам (5) и (6).

Определение начальной интенсивности отказов компьютерной программы.

В соответствии с [2] начальную интенсивность проявления ошибок (интенсивность отказов) компьютерной программы λ_0 рекомендуется определять по модели

$$\lambda_0 = 60 \frac{R}{B} K_\Sigma F_0 L \cdot 10^{-6}, \text{ч}^{-1}, \quad (9)$$

где R – пиковое быстродействие процессора компьютера, указываемое производителем в технической документации, подставляется в размерности «операций в секунду».

Коэффициент K_Σ выбирается по таблице 1 в зависимости от области применения (назначения) компьютерной программы. Начальная плотность отказов F_0 определяется по (8), значение L (в строках кода – LOC) прогнозируется при подготовке исходных данных.

Определение процессорного времени тестирования для обеспечения заданной эксплуатационной надёжности компьютерной программы.

На этом этапе вначале необходимо определить требуемое значение коэффициента эффективности тестирования. Его находят как

$$Q = \frac{\lambda_{\text{экс}}}{\lambda_0}, \tag{10}$$

где $\lambda_{\text{экс}}$ – заданная эксплуатационная интенсивность отказов компьютерной программы (интенсивность проявления ошибок).

С учётом [3] требуемое процессорное временя тестирования определится по формуле

$$t = \frac{B \cdot \ln(Q)}{60 K_\Sigma R} 10^6, \text{ ч}, \tag{11}$$

в которую пиковое быстродействие процессора R следует подставлять в размерности «операций в секунду».

Для прикладных компьютерных программ прогнозное значение рабочей календарной продолжительности тестирования $T^{(i)}_{\text{календ}}$, соответствующей заданному процессорному времени тестирования t, получаемому по формуле (11), примерно может быть получено по модели

$$T^{(i)}_{\text{календ}} = \frac{100\,t}{r^{(i)}}, \text{ ч}, \tag{12}$$

где верхний индекс (i) указывает на то, что соответствующие характеристики относятся к прикладным компьютерным программам i-й области (отрасли) применения; $r^{(i)}$ – средний процент времени выполнения (прогона) компьютерной программы при её тестировании в течение календарной продолжительности для прикладных компьютерных программ i-й области применения.

Заключение.

Предлагаемая методика позволяет оценить ожидаемую начальную надёжность планируемой к разработке прикладной компьютерной программы и спрогнозировать планируемое календарное время тестирования, необходимое для обеспечения заданной эксплуатационной надёжности компьютерной программы. Решение этих задач имеет важное значение для проектирования информационно-компьютерных систем, в том числе используемых для технологий Big Data, и позволяет ещё до разработки прикладной компьютерной программы, предназначенной для её использования в составе сложной информационной системы, получить представление о предполагаемых затратах времени и средств для обеспечения заданной эксплуатационной надёжности компьютерной программы.

Список использованных источников

[1] Чуканов, В. О. Методы обеспечения аппаратно-программной надёжности вычислительных систем / В. О. Чуканов, В. В. Гуров, Е. В. Прокопьева [Электронный ресурс]. – 2014. – Режим доступа : http://www.mcst.ru/metody-obespecheniya-apparatnoprogrammnoj-nadezhnosti-vychislitelnykh-sistem. – Дата доступа: 10.03.2022.

[2] Borovikov S. M., Kaziuchyts V. O., Khoroshko V. V., Dick S. S., Klinov K. I. Assessment of expected reliability of applied software for computer-based information systems. Informatics, 2021, vol. 18, no. 1, pp. 84–95 (in Russian). https://doi.org/10.37661/1816-0301-2021-18-1-84-95.

[3] Borovikov S. M., Kaziuchyts V. O., Dick S. K., Dick S. S. Модель прогнозирования времени тестирования прикладных компьютерных программ для технологий BIG DATA / BIG DATA and Advanced Analytics = BIG DATA и анализ высокого уровня : сборник научных статей VII Международной научно-практической конференции, Минск, 19-20 мая 2021 года / редкол.: В. А. Богуш [и др.]. – Минск : Бестпринт, 2021. – С. 404-411.

[4] Боровиков С. М., Дик С. С., Лэ В. Т., Клинов К. И. Анализ и оценка надёжности прикладных компьютерных программ / BIG DATA and Advanced Analytics = BIG DATA и анализ высокого уровня: сб. материалов VI Междунар. науч.-практ. конф., Минск, 20-21 мая 2020 года: в 3 ч. Ч. 1 / редкол. : В. А. Богуш [и др.]. – Минск : Бестпринт, 2020. – С. 382-390.

[5] Боровиков, С. М. Модель прогнозирования ожидаемой надёжности прикладных компьютерных программ / Боровиков С. М., Казючиц В. О. // Информационные радиосистемы и радиотехнологии 2020 : материалы Республиканской научно-практической конференции, Минск, 28-29 октября 2020 г. / Белорусский государственный университет информатики и радиоэлектроники ; редкол.: В. А. Богуш [и др.]. – Минск : БГУИР, 2020. – С. 292-295.

[6] Боровиков С. М., Дик С. С., Лэ В. Т., Клинов К. И. Возможный подход к оценке надёжности разрабатываемых программных средств на ранних этапах проектирования информационно-компьютерных систем / Globus: технические науки – от теории к практике. – 2020. – Вып. 1 (32). – С. 4–9.

[7] Модель прогнозирования надёжности планируемых к разработке прикладных компьютерных программ / Интернаука: научный журнал. – 2020. – № 12 (141). – Ч. 1. – С. 68-72.

[8] Боровиков, С. М. Метод прогнозирования надёжности прикладных программных средств на ранних этапах их разработки / С. М. Боровиков, С. С. Дик, Н. К. Фоменко // Доклады БГУИР. – 2019. – № 5 (123). – С. 45-51. - DOI: http://dx.doi.org/10.35596/1729-7648-2019-123-5-45-51.

[9] Боровиков С. М., Будник А. В., Дик С. С, Лэ В. Т. Подход к оценке ожидаемой надёжности прикладного программного обеспечения для систем телекоммуникаций / Современные средства связи : материалы XXIV Междунар. науч.-техн. конф., Минск, 17–18 окт. 2019 г. // редкол. : А. О. Зеневич [и др.]. – Минск : БГАС, 2019. – С. 120-122.

[10] Боровиков, С. М. Возможный подход к оценке надёжности прикладных программных средств для технологий Big Data / С. М. Боровиков, Ван Там Лэ, С. С. Дик // BIG DATA and Advanced Analytics = BIG DATA и анализ высокого уровня : сб. материалов V Междунар. науч.-практ. конф. (Республика Беларусь, Минск, 13–14 марта 2019 года). В 2 ч. Ч. 2 / редкол. : В. А. Богуш [и др.]. – Минск : БГУИР, 2019. – С. 77-83.

[11] Боровиков, С. М. Прогнозирование ожидаемой надёжности прикладных программных средств с использованием статистических моделей их безотказности / С. М. Боровиков, С. С. Дик // BIG DATA Advanced Analytics: collection of materials of the fourth international scientific and practical conference, Minsk, Belarus, May 3 – 4, 2018 / editorial board: M. Batura [etc.]. – Minsk : BSUIR : 2018. – P. 348-354.

[12] Боровиков, С. М. Метод оценки ожидаемой надёжности прикладных компьютерных программ систем медицинской электроники / С. М. Боровиков, С. С. Дик // Доклады БГУИР. – 2018. – № 7 (117). – С. 112-117.

[13] Лэ, В. Т. Метод оценки надёжности прикладных программных средств на ранних этапах их разработки / В. Т. Лэ, С. С. Дик, С. М. Боровиков // Современные средства связи : материалы XXIII Междунар. науч.-техн. конф., Минск, 18–19 окт. 2018 года // редкол. : А. О. Зеневич [и др.]. – Минск : БГАС, 2018. – С. 167-168.

[14] McCall, J. A. Software Reliability, Measurement, and Testing Guidebook for Software Reliability Measurement and Testing [Electronic resource] / J. A. McCall [et al.]. – 1992. – Mode of access: https://apps.dtic.mil/dtic/tr/fulltext/u2/a256164.pdf. – Date of access: 10.03.2022.

[15] ГОСТ 27.205-97. Надёжность в технике. Проектная оценка надёжности сложных систем с учётом технического и программного обеспечения и оперативного персонала. Основные положения. – Минск : Госстандарт Республики Беларусь, 2005. – 22 с.

[16] Шубинский, И. Б. Функциональная надёжность информационных систем. Методы анализа / И. Б. Шубинский. – М. : «Журнал Надёжность», 2012. – 296 с.

[17] Постановление министерства труда и социальной защиты Республики Беларусь 27 июня 2007 г. № 91 «Об утверждении укрупнённых норм затрат труда на разработку программного обеспечения»

[Электронный ресурс]. – Режим доступа : https:// https://zakonrb.com/npa/ob-utverzhdenii-ukrupnennyh-norm-zatrat-truda (дата обращения: 12.03.2022).

[18] Чуканов, В. О. Надёжность программного обеспечения и аппаратных средств систем передачи данных атомных электростанций : учебное пособие / В. О. Чуканов. – М. : МИФИ, 2008. – 168 с.

METHODOLOGY FOR ENSURING THE OPERATIONAL RELIABILITY OF APPLIED COMPUTER PROGRAMS, PLANNED TO DEVELOPMENT FOR INFORMATION SYSTEMS

S.M. BOROVIKOV
PhD
Associate professor of the BSUIR

V.O. KAZIUCHYTS
Master of engineering
Postgraduate student of the BSUIR

S.K. DICK
PhD
Associate professor of the BSUIR

S.S. DICK
Master of Engineering and Technology
Project manager of "Itransishen" Company

Belarusian State University of Informatics and Radioelectronics, Republic of Belarus
Itransition Company, Republic of Belarus
E-mail: bsm@bsuir.by

Abstract. Applied computer programs used for information systems, including Big Data technologies, contain tens of thousands to millions of lines of program code, and therefore, after writing, they always have hidden errors that cannot be determined by the programming language compiler. The task of the testing phase is to identify and eliminate the most critical errors in terms of the performance of the computer program of its functions and reduce the number of hidden errors remaining in the program. Hidden errors that remain in the program after testing it can sometimes manifest themselves depending on the degree of variability of the initial data and the load that the operating environment has on the computer program (writing information on electronic media, printing on a printer, waiting in line, etc.). The presence of these hidden errors determines the level of operational reliability of computer programs. Developers of information systems, including analytical systems used in Big data technologies, would like to know how much time and, consequently, financial costs will be required for testing application computer programs planned for development in order to ensure a given level of their operational reliability. The paper explains the proposed methodology, using which you can get an answer to these questions. The technique was developed on the basis of experimental data on the reliability of applied computer programs for various functional purposes.

Keywords: applied computer programs, initial reliability, ensuring operational reliability, testing efficiency, testing time prediction.