DOI: 10.61726/6446.2025.12.87.001
УДК 004.4' 242:004.6-027.45

НАДЕЖНОСТЬ РАЗРАБАТЫВАЕМЫХ ПРИКЛАДНЫХ КОМПЬЮТЕРНЫХ ПРОГРАММ ДЛЯ ИНФОРМАЦИОННЫХ СИСТЕМ

С.М. БОРОВИКОВ[1], А.В. БУДНИК[2], В.Т. ЛЭ[3]

[1]*Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»,*
ул. П. Бровки, 6, 220013, Минск, Беларусь
ORCID: https://orcid.org/0000-0002-3398-9500

[2]*Учреждение образования «Белорусская государственная академия связи»,*
ул. Ф. Скорины, 8/2, Минск, 220076, Беларусь
ORCID: https://orcid.org/0009-0003-8735-3852

[3]*Ханойский технологический институт,*
ул. Кау Вонг, № 3, Донг Нгак, Ханой, Вьетнам
ORCID: https://orcid.org/0009-0003-3164-0211

Поступила в редакцию 21 апреля 2025

Цель работы состоит в рассмотрении подходов к оценке ожидаемой эксплуатационной надежности прикладных компьютерных программ, используемых в информационных системах. В работе приводятся подготовленные рекомендации, предназначенные для оценки безотказности планируемых к разработке компьютерных программ для информационных систем разного функционального назначения. Использование рекомендаций позволит разработчикам оценивать прогнозный уровень надежности компьютерных программ и более достоверно определять проектную эффективность функционирования систем.

Ключевые слова: компьютерные информационные системы, прикладные компьютерные программы, ожидаемая надежность программ, тестирование.

Введение. Развитие и совершенствование технологий Big Data обуславливают необходимость разработки новых прикладных компьютерных программ по обработке больших объемов данных. Успех в обработке больших объемов данных зависит от надежности используемых прикладных компьютерных программ. Для принятия решения о целесообразности разработки и уровне эффективности функционирования компьютерной информационной системы в заданных условиях необходимо на ранних этапах ее проектирования оценить ожидаемую эксплуатационную надежность планируемых к разработке прикладных компьютерных программ. Под эксплуатационной надежностью будем понимать тот уровень надежности, который компьютерная программа покажет на начальном этапе ее эксплуатации, то есть после выполнения этапа тестирования. В настоящее время проблема обеспечения надежности разрабатываемых компьютерных программ стоит остро, поскольку вклад программного обеспечения в ненадежность сложных компьютерных информационных систем может составлять до 40 и более процентов [1, 2]. В данной работе на основе систематизации положений, приводимых в ранее выполненных работах [1, 3–13], излагаются основные принципы оценки ожидаемой надежности и приводятся проектные процедуры по определению ожидаемого показателя безотказности планируемых к разработке прикладных компьютерных программ.

Основные принципы и подходы. Ниже указаны основные принципы и подходы, принимаемые во внимание при оценке ожидаемой надежности планируемой к разработке прикладной компьютерной программы.

1. Специалисты по программированию пришли к выводу, что при оценке надежности компьютерных программ в качестве единственно приемлемой количественной характеристики

измерения объема (размера) программы является число исполняемых строк программного кода (в англоязычном варианте LOC – Lines Of Code) [14].

2. Размер (объем) прикладных компьютерных программ современных информационных систем составляет сотни тысяч и даже миллионы строк программного кода. Как пример, в табл. 1 приводится объем программного обеспечения, используемого на некоторых объектах.

Таблица 1

Объем программного обеспечения

Компьютерная программа (объект)	Космическая станция	Космический корабль	Самолет «Boeing 777»	Операционная система Windows XP для ПК
Примерное количество строк кода, млн	40	10	7	40

По своей природе программное обеспечение сложнее технических средств систем. Объем программных средств для современных информационно-компьютерных систем оценивается в 1…100 млн и более команд или информационных слов. Вклад программного обеспечения в ненадежность сложных компьютерных информационных систем может превышать вклад, вносимый техническими средствами (компьютерами), поскольку входные данные могут быть сложнее и их формат все время меняется [14]. Надежность аппаратуры ограничивается ошибками проектирования, производственными дефектами и частотой сбоев (зависит от физических процессов).

3. Формы проявления ненадежности прикладных компьютерных программ разнообразны: «зависание» программы, переполнение памяти, невозможность продолжения выполнения из-за некорректности входных данных, деление на нуль и т. д.

Ненадежность прикладной компьютерной программы большого размера (объема) является следствием наличия в ней скрытых неявных ошибок, оставшихся после выполнения процедуры тестирования в течение отводимого времени на разработку программы. В больших компьютерных программах (объем: тысячи–десятки тысяч и более строк кода) всегда остаются скрытые ошибки, 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) – $\lambda_{экс}$. Задача тестирования состоит в том, чтобы выявить и устранить наиболее критичные ошибки (ошибки с точки зрения функциональности программы), свести к минимуму долю скрытых ошибок, оставшихся в компьютерной программе, и в итоге обеспечить приемлемый уровень эксплуатационной интенсивности отказов $\lambda_{экс}$.

Среднее число ошибок, приходящихся на тысячу строк кода (KLOC) варьируется от 5 до 50 ошибок [14]. Для ответственного программного обеспечения (ПО), к которому можно отнести операционные системы и другое системное ПО, к моменту поставки системы клиенту в нем может содержаться 0,04...0,15 ошибок на 1000 строк кода программы.

В прикладных компьютерных программах, которые протестированы только на предмет обеспечения функциональных возможностей, присутствует около 30…50 ошибок на 1000 выполняемых строк программного кода (рис. 1).

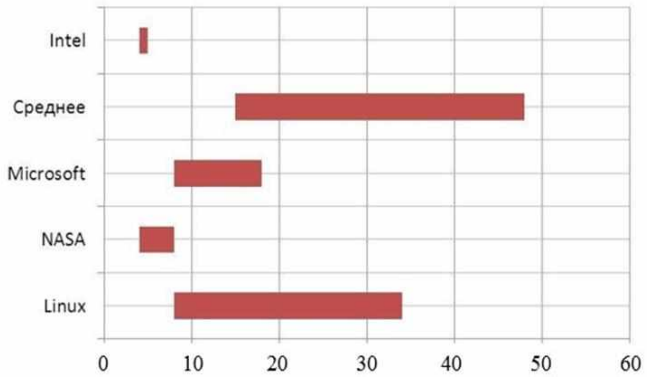

Рис. 1. Среднее число ошибок, приходящихся на 1000 строк кода (KLOC), для прикладных компьютерных программ, прошедших тестирование

4. Интерес представляет модель надежности прикладной компьютерной программы, учитывающая [1, 4]:

– 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 ее модулей из памяти и т. д.);

– особенность организации, которая будет разрабатывать программу;

– основные характеристики программы: сложность, новизна, средства разработки, использование стандартных модулей;

– быстродействие процессора в составе используемого компьютера.

5. Эксплуатационная интенсивность отказов ($\lambda_{экс}$) характеризует надежность компьютерной программы, но значение этой характеристики зависит от быстродействия процессора, используемого в составе компьютера. Поэтому с учетом документа [15], в качестве основной характеристики безотказного выполнения программой своих функций рекомендуется использовать вероятность того, что прикладная компьютерная программа безотказно выполнит обработку одного произвольного набора исходных данных из числа тех наборов, которые могут поступать в условиях использования компьютерной программы в составе информационной системы. Далее эта вероятность обозначена через $p_1(t_1)$, где t_1 – среднее процессорное время обработки компьютерной программой одного набора исходных данных. Значение вероятности $p_1(t_1)$ практически не зависит от быстродействия процессора (компьютера).

Исходные данные для оценки ожидаемой надежности планируемой к разработке прикладной компьютерной программы. На раннем этапе разработки прикладной компьютерной программы в качестве исходных данных должны рассматриваться:

1. Область (сфера, отрасль) применения программы.
2. Конкретные функции, выполняемые программой.
3. Предполагаемый язык написания программного кода.
4. Прогнозируемое число строк программного кода.

Оценить общий прогнозный объем будущей компьютерной программы в выполняемых строках кода (L) можно, например, исходя из количества и объема возлагаемых на программу функций [17]:

$$L = \sum_{i=1}^{n} v_i \ , \qquad (1)$$

где v_i – объем в строках кода (LOC), необходимый для выполнения компьютерной программой i-й функции; n – число функций, возлагаемых на программу.

В случае неопределенности значения v_i могут быть взяты из каталога функций программного обеспечения, включенных в документ [17]. В каталоге приводятся обобщенные (усредненные) значения v_i в зависимости от среды разработки компьютерных программ и кода функции. Например, для функции с кодом 207 – «Организация поиска и поиск в базе данных» при использовании Visual C++ (Microsoft) v_i = 5480 LOC, а для функции с кодом 705 – «Формирование и вывод на внешние носители документов сложной формы» при использовании языка Java v_i = 3150 LOC.

Конечное значение L следует представлять в тысячах строках кода (англоязычный вариант kilolines of code – KLOC), выполняя округление в большую сторону.

5. Характеристика производственной среды разработки программного обеспечения, в том числе:

а) особенность организации, разрабатывающей компьютерную программу;

б) квалификация программистов и их опыт.

6. Характеристика планируемой к разработке компьютерной программы, в том числе следующие ее особенности:

а) степень сложности компьютерной программы;

б) степень новизны компьютерной программы;

в) используемые средства разработки компьютерной программы;

г) примерный процент использования стандартных модулей в будущей компьютерной программе.

7. Календарное рабочее время (в часах), планируемое на выполнение процедуры тестирования программы, после написания кода программы и устранения ошибок, обусловленных нарушением правил языка программирования.

8. Заданный календарный период времени (в часах), для которого интересуются безотказностью компьютерной программы (обозначим этот период через τ), и интенсивность обращения к компьютерной программе в течение периода τ. Интенсивность представляет собой среднее число сеансов («прогонов») программы за один час в процессе функционирования компьютерной информационной системы в течение календарного периода τ. Обозначим эту интенсивность через η, ее размерность – 1/ч.

9. Пиковое быстродействие процессора, используемого в составе компьютера. Далее обозначено через R, размерность – операций в секунду.

10. Суммарный бюджет рабочего времени (в часах), отводимого на тестирование программы, обозначено через $T_Б$.

Последовательность получения прогнозной оценки вероятности безотказной обработки разрабатываемой компьютерной программой одного произвольного набора исходных данных.

Этап А. *Начальная интенсивность проявления ошибок (интенсивность отказов) компьютерной программы* – λ_0.

1. Используемая модель оценки λ_0 [2]

$$\lambda_0 = 60\frac{R}{B}K_\Sigma F_0 L\cdot 10^{-6}, \text{ч}^{-1}, \quad (2)$$

где B – среднее число команд, которое будет выполняться программой в течение одного прогона для экстремального набора входных данных и рабочей нагрузки на компьютерную программу со стороны эксплуатационной среды; K_Σ – коэффициент увеличения интенсивности отказов из-за суммарного действия изменчивости входных данных и рабочей нагрузки на компьютерную программу со стороны эксплуатационной среды (ввод данных, использование принтера, ожидание операций в очереди и т. д); F_0 – прогнозируемая начальная плотность ошибок в компьютерной программе, до выполнения тестирования программы.

Размерности параметров формулы (2): [R] – операций/секунда; [B] – операций (команд, инструкций); [F_0] – ошибка/KLOC; [L] – KLOC.

2. Коэффициент K_Σ. Выбирается в зависимости от области применения будущей компьютерной программы (табл. 2).

Таблица 2

Выбор коэффициентов для прикладных компьютерных программ различных областей применения

Область применения компьютерной программы	Значение $F_Б$, ошибка/KLOC	Коэффициент K_Σ	Средний процент времени «прогона» компьютерной программы при ее тестировании в течение рабочего дня
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	8,7	8,83	12

Примечание. Область применения, наиболее близкая к обработке Big Data.

3. Определение F_0. В соответствии с «Римской моделью» RL-92-52 [17]:

$$F_0 = F_Б \cdot D \cdot S \text{ , ошибка/KLOC,} \quad (3)$$

где метрика $F_Б$ – базовая (усредненная) плотность ошибок, выбираемая по табл. 2 в зависимости от области применения компьютерной программы; D – метрика, учитывающая характеристику производственной среды разработки программы; S – метрика, учитывающая характеристики разрабатываемой компьютерной программы.

4. Определение метрики D. В простейшем случае ее определяют, как произведение коэффициентов, учитывающих:

а) особенность проектной организации ($K_{орг}$);

б) квалификацию и опыт программистов ($K_{квал}$):

$$D = K_{орг} \cdot K_{квал} . \quad (4)$$

Выбор коэффициента $K_{орг}$ может быть сделан по табл. 3, либо методом экспертной оценки из условия $0{,}5 \le D \le 2$, причем $D = 0{,}5$ – наиболее благоприятные условия, а $D = 2$ – удовлетворительные условия для обеспечения качества написания программного кода.

Таблица 3

Коэффициент $K_{орг}$ для метрики производственной среды D

Характеристика организации, разрабатывающей компьютерную программу	Значение $K_{орг}$
1. Группа программистов работает в организации, эксплуатирующей компьютерную информационную систему	0,76
2. Группа программистов имеет опыт работы по разработке компьютерных программ, но не связана с организацией, эксплуатирующей систему	1,00
3. Группа программистов обладает опытом работы с компьютером, но не знакома с ПО, для которого разрабатывается программа, и используемым комплексом технических средств и действиями операторов	1,30

Для выбора коэффициента $K_{квал}$ рекомендуется использовать данные табл. 4 [1, 10, 17].

Таблица 4

Значения коэффициента $K_{квал}$

Квалификация и опыт программиста	Значение $K_{квал}$
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

5. Определение метрики S. В простейшем случае метрика может быть определена как произведение коэффициентов, учитывающих:

а) степень сложности компьютерной программы ($K_{слож}$);

б) средства разработки компьютерной программы ($K_{ср.разр}$)

в) степень новизны компьютерной программы ($K_{нов}$);

г) процент использования стандартных модулей в компьютерной программе ($K_{мод}$):

$$S = K_{слож} \cdot K_{нов} \cdot K_{ср.разр} \cdot K_{мод} . \quad (5)$$

Выбор коэффициента $K_{слож}$. В соответствии с [16] значение коэффициента следует определять по формуле

$$K_{слож} = 1 + \sum_{i=1}^{m} k_i , \quad (6)$$

где k_i – коэффициент повышения сложности прикладной компьютерной программы за счет наличия у нее i-й характеристики из числа, приведенных в табл. 5; m – число характеристик повышения сложности, присущих планируемой к разработке прикладной компьютерной программы.

Таблица 5

Коэффициенты повышения сложности компьютерной программы (k_i)

Характеристика повышения сложности компьютерной программы		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 сложности из числа, указанных в таблице 6 (из пп. 4.1–4.3 для подстановки в формулу (6) выбирается одно из значений k_i)	4.1 Две характеристики	0,12
	4.2 Три характеристики	0,18
	4.3 Свыше трех характеристик	0,26

В табл. 6 приведено описание характеристик сложности прикладных компьютерных программ. Следует уточнить, какими характеристиками должна обладать планируемая к разработке компьютерная программа. Это позволит обоснованно выбрать коэффициент повышения сложности k_i, рассматривая п. 4 табл. 5.

Таблица 6

Описание характеристик сложности прикладных компьютерных программ (для п. 4 табл. 5)

Номер характеристики	Описание характеристики компьютерной программы
1	Наличие сложного интеллектуального языкового интерфейса с пользователем
2	Обеспечение телекоммуникационной обработки данных и управление удаленными 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 расчетов

Выбор коэффициента $K_{нов}$. При определении значения коэффициента можно воспользоваться данными табл. 7 [16].

Таблица 7

Коэффициент $K_{нов}$, учитывающий новизну компьютерной программы

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 определенного параметрического ряда компьютерных программ	+	+	1,0
	–	+	0,81
	+	–	0,72
Компьютерные программы, являющиеся развитием определенного параметрического ряда компьютерных программ, разработанных для ранее освоенных типов конфигурации ПК и ОС	–	–	0,63

* Принятые сокращения: ПК – персональный компьютер, ОС – операционная система

Выбор коэффициента $K_{ср.разр}$. Рекомендуется использовать значения, приведенные в табл. 8 [16].

Таблица 8

Коэффициент $K_{ср.разр}$, учитывающий средства разработки

Средства разработки компьютерной программы	Значение коэффициента $K_{ср.разр}$ в зависимости от ОС		
	IBM-PC, Windows	Функционирование программы в сетях	
		локальных	глобальных
Языки высокого уровня (C++, Pascal и др.)	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

Выбор коэффициента $K_{мод}$. Следует использовать данные табл. 9 [16].

Таблица 9

Коэффициент $K_{мод}$, учитывающий процент использования стандартных модулей

Степень охвата стандартными модулями реализуемых функций	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	1

Определение количества команд компьютерной программы. Прогнозное количество команд программы (B) может быть найдено как произведение

$$B = L\,E_L\,E_{ц}, \tag{7}$$

где E_L – коэффициент расширения кода программы относительно числа строк кода L, определяется используемым языком программирования; $E_{ц}$ – коэффициент увеличения числа выполняемых процессором команд за счет наличия в компьютерной программе условных переходов, ветвлений и других особенностей (ввод-вывод данных, считывание информации с внешних носителей, ожидание в очереди, обработка прерываний и т. д.).

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

Значение $E_{ц}$ следует выбирать на основе экспертной оценки с учетом алгоритма выполнения компьютерной программой своих функций, экстремального характера входных данных, числа ветвлений в программе, записи и считывания данных с внешних носителей и т. п. В зависимости от особенностей программы и реальных потоков набора исходных данных, поступающих в программу, значение $E_{ц}$ может достигать десятков–сотен единиц.

Этап Б. *Эксплуатационная интенсивность проявления ошибок (интенсивность отказов) компьютерной программы* – $\lambda_{экс}$.

Прогнозное значение $\lambda_{экс}$ следует рассчитывать с учетом проектного значения коэффициента тестирования Q [1] компьютерной программы, определяемого в предположении, что тестирование программы выполняется в течение заданного (бюджетного) календарного рабочего времени $T_Б$. Согласно работам [8, 12, 13],

$$Q = \exp\left[\frac{60 \cdot 10^{-6} K_{\Sigma}\, R\, T_{Б}\, (r/100)}{B}\right], \tag{8}$$

где r – средний процент времени выполнения (прогона) компьютерной программы при ее тестировании в течение календарного рабочего времени для прикладных компьютерных программ рассматриваемой области применения (табл. 2).

Размерность параметров, используемых в формуле (8): [R] – операций/секунда; [B] – операций (команд, инструкций); [$T_Б$] – ч; [r] – процент (%).

Принимая во внимание, что $Q = \lambda_0 / \lambda_{экс}$ [1], окончательно будем иметь

$$\lambda_{экс} = \frac{\lambda_0}{Q}. \tag{9}$$

Этап В. *Вероятность того, что оставшиеся после тестирования скрытые ошибки в компьютерной программе не проявятся в течение заданного календарного рабочего времени τ.*

Вначале определяют вероятность того, что компьютерная программа безошибочно выполнит обработку одного произвольного набора исходных данных из числа наборов, которые могут поступать в условиях функционирования компьютерной программы. Согласно работе [19], для расчета этой вероятности (p_1) используют экспоненциальную функцию:

$$p_1(t_1) = \exp\left(\lambda_{экс} \cdot t_1\right), \tag{10}$$

где t_1 – среднее процессорное время обработки компьютерной программой одного набора исходных данных, определяемое как

$$t_1 \approx \frac{B}{3600\,(0{,}7 \cdot R)}, \text{ ч.} \tag{11}$$

В формулу (11) значение R необходимо подставлять в размерности «операций в секунду».

Вероятность того, что оставшиеся после тестирования скрытые ошибки в компьютерной программе не проявятся в течение заданного календарного рабочего времени τ, следует определять по формуле [15]

$$P(\tau) = \left(p_1\right)^{\eta \cdot \tau}. \tag{12}$$

Если значение полученной вероятности $P(\tau)$ не отвечает требованиям заказчика, то рекомендуется предпринять меры по увеличению вероятности $p_1(t_1)$ путем изменения заданного времени тестирования компьютерной программы ($T_Б$). Новое время $T_Б$ должно быть согласовано с заказчиком. С учетом скорректированного времени тестирования программы $T_Б$ следует по формуле (8) пересчитать прогнозное значение коэффициента тестирования Q, а затем по отношению (9) найти уточненное значение ожидаемой эксплуатационной интенсивности ($\lambda_{экс}$) проявления программой скрытых ошибок. После чего по формулам (10) и (12) необходимо получить новые значения интересующих вероятностей $p_1(t_1)$ и $P(\tau)$.

Заключение. Предложенный подход дает приближенные результаты ожидаемого уровня эксплуатационной надежности планируемой к разработке прикладной компьютерной программы. Однако даже такой ориентировочный расчет полезен, так как дает возможность получить представление о надежности компьютерной программы на раннем этапе ее разработки, до написания программного кода. Это позволит для создаваемой компьютерной информационной системы оценить проектный уровень эффективности функционирования. Кроме того, используя описанный подход к оценке надежности разрабатываемой прикладной компьютерной программы, можно определить примерное рабочее время ее тестирования, необходимое для обеспечения заданного уровня надежности.

RELIABILITY OF DEVELOPED APPLICATION COMPUTER PROGRAMS OF INFORMATION SYSTEMS

S.M. BOROVIKOV, A.V. BUDNIK, V.T. LE

Abstract

The purpose of the work is to consider approaches to assessing the expected operational reliability of applied computer programs used in information systems. The work provides prepared recommendations intended to assess the reliability of planned computer programs for information systems of various functional purposes. The use of recommendations will allow developers to assess the predicted level of reliability of computer programs and more reliably determine the design efficiency of the functioning of the systems.

Список литературы

1. Assessment of expected reliability of applied software for computer-based information systems / S. M. Borovikov [et al.] // Informatics. – 2021. – Vol. 18, № 1. – P. 84–95.

2. Чуканов, В. О. Методы обеспечения аппаратно-программной надежности вычислительных систем / В. О. Чуканов, В. В. Гуров, Е. В. Прокопьева [Электронный ресурс]. – Режим доступа : http://www.mcst.ru/files/5357ec/dd0cd8/50af39/000000/seminar_metody_obespecheniya_apparatnoprogrammnna dezhnosti_vychislitelnyh_sistem.pdf. – Дата доступа : 24.02.2025.

3. Боровиков, С. М. Возможный подход к оценке надежности прикладных программных средств для технологий Big Data / С. М. Боровиков, В. Т. Лэ, С. С. Дик // BIG DATA и анализ высокого уровня: сб. матер. V Международ. науч.-практ. конф. / Минск (13–14 марта 2019 г.). – Ч. 2. – Минск, 2019. – С. 77–83.

4. Боровиков, С. М. Анализ и оценка надежности прикладных компьютерных программ / С. М. Боровиков, В. Т. Лэ, К. И. Клинов // BIG DATA и анализ высокого уровня: сб. матер. VI Междунар. науч.-практ. конф. / Минск (20–21 мая 2020 г.). – Ч. 1. – Минск : Бестпринт, 2020. – С. 382–390.

5. Боровиков, С. М. Модель прогнозирования надежности планируемых к разработке прикладных компьютерных программ / С. М. Боровиков, С. С. Дик, В. Т. Лэ, К. И. Клинов // Интернаука. – 2020. – № 12 (141), Ч. 1. – С. 68–72.

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

7. Боровиков, С. М. Модель прогнозирования ожидаемой надежности прикладных компьютерных программ / С. М. Боровиков, В. О. Казючиц // Информационные радиосистемы и радиотехнологии 2020: Матер. Республ. науч.-практ. конф. / Минск (28–29 октября 2020 г.). – Минск: БГУИР, 2020. – С. 292–295.

8. Боровиков, С. М. Модель прогнозирования времени тестирования прикладных компьютерных программ для технологий BIG DATA / С. М. Боровиков [и др.] / BIG DATA и анализ высокого уровня : сб. науч. статей VII Междунар. науч.-практ. конф. / Минск (19–20 мая 2021 г.). – Минск: Бестпринт, 2021. – С. 404–411.

9. Боровиков, С. М. Модель прогнозирования времени тестирования прикладных компьютерных программ для автоматизированных систем управления / С. М. Боровиков [и др.] // Информационные технологии и системы 2021: Матер. Международ. науч. конф. ИТС-2021 / Минск (24 ноября 2021 г.). – Минск, 2021. – С. 28–29.

10. Боровиков, С. М. Методика обеспечения эксплуатационной надежности планируемых к разработке прикладных компьютерных программ для информационных систем / С. М. Боровиков [и др.] // BIG DATA и анализ высокого уровня: сб. науч. статей VIII Международ. науч.-практ. конф. / Минск (11–12 мая 2022 г.). – Минск: Бестпринт, 2022. – С. 162–173.

11. Казючиц, В. О. Модель прогнозирования времени тестирования компьютерной программы автоматизированной оценки надежности полупроводниковых приборов / В. О. Казючиц, С. М. Боровиков, Е. Н. Шнейдеров // Доклады БГУИР. – 2022. – Т. 20, № 7. – С. 72–80.

12. Лэ, В. Т. Рекомендации по оценке и обеспечению надежности планируемых к разработке прикладных компьютерных программ для информационных систем / В. Т. Лэ [и др.] // BIG DATA и анализ высокого уровня: сб. науч. статей IX Международ. науч.-практ. конф. / Минск (17–18 мая 2023 г.). – Ч. 2. – Минск, 2023. – С. 78–86.

13. Лэ, В. Т. Ускоренная оценка надежности разрабатываемых прикладных компьютерных программ / В. Т. Лэ [и др.] // BIG DATA и анализ высокого уровня: Сб. науч. статей X Междунар. науч.-практ. конф. / Минск (13 марта 2024 г.). – Минск, 2024. – С. 444–451.

14. Программное обеспечение – источник всех проблем [Электронный ресурс]. – Режим доступа : http://www.williamspublishing.com/PDF/5-8459-0785-3/part1.pdf. – Дата доступа : 24.02.2025.

15. ГОСТ 27.205-97 Надежность в технике. Проектная оценка надежности сложных систем с учетом технического и программного обеспечения и оперативного персонала. Основные положения. – Минск: Госстандарт Республики Беларусь, 2005. – 22 с.

16. Постановление министерства труда и социальной защиты Республики Беларусь 27 июня 2007 г. № 91 «Об утверждении укрупненных норм затрат труда на разработку программного обеспечения» [Электронный ресурс]. – Режим доступа : https://news-newsby-org.narod.ru/doc-razn/pos05/postn05236/index.htm. – Дата доступа : 24.02.2025.

17. McCall J.A., Randell W., Dunham J., Lauterbach L. Software reliability, measurement, and testing guidebook for software reliability measurement and testing // RL-TR-92-52: Final Technical Report, Vol II (of two) / Science Applications International Corp., Research Triangle Institute. – Rome Laboratory, NY 13441-5700, 1992. – 256 p. – Режим доступа : https://apps.dtic.mil/sti/tr/pdf/ADA256164.pdf. – Дата доступа : 24.02.2025.

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

19. Шубинский, И. Б. Функциональная надежность информационных систем. Методы анализа / И. Б. Шубинский. – М. : Журнал Надежность, 2012. – 296 с.