УДК 004.519.217

ДОСЛІДЖЕННЯ ТОЧНОСТІ ОЦІНЮВАННЯ НАДІЙНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Д. А. Маєвський, О. В. Найденко, О. Ю. Маєвська, О. В. Стрельцов, А. А. Найденко
Одеський національний політехнічний університет

Анотація. *Наведено залежність точності оцінювання надійності програмного забезпечення від мови програмування і моделі надійності. Виконано моделювання надійності для декількох програмних систем за допомогою основних моделей. Встановлено, що точність оцінювання надійності програмного забезпечення не залежить від мови програмування.*

Ключові слова: *точність оцінювання, програмне забезпечення, модель надійності, мова програмування, виявлення дефектів, відмова.*

Вступ

Розрахунок надійності програмної продукції є складним завданням, хоча і дуже актуальним завдяки надзвичайному поширенню цифрової техніки в усіх галузях життєдіяльності людини. Поряд з незаперечними перевагами використання програмного забезпечення (ПЗ), його невірне функціонування може призвести як до серйозних матеріальних збитків, так і до загрози життю людей та навколишнього середовища[1].

На сьогодні немає навіть однозначного трактування основних термінів надійності програм. Питання статистичних досліджень програм є складним і через те, що їх тиражування не таке, як апаратних систем. З плином часу зменшується кількість програмних відмов і збільшується час між цими відмовами завдяки тому, що програмісти при тривалому часі роботи набувають певних навичок, підвищують свою кваліфікацію і успішно локалізують як власне помилки, так і зони їх можливих появ.

Оцінка показників надійності ПЗ ґрунтується на результатах тестування програмної системи з використанням математичних моделей надійності програмного забезпечення (МНПЗ). У літературі описано велику кількість МНПЗ, а також їх різні варіанти і комбінації. Так, за даними [2], на сьогодні створено більше ніж шістдесят різних моделей та їх модифікацій. Однак велика кількість моделей створює проблему вибору найбільш придатної моделі для даної програмної системи в залежності від її предметної області і мови програмування, а також роблять актуальним дослідження МНПЗ з позиції точності оцінювання надійності програмного забезпечення.

Слід визнати, що абсолютно надійних програм не існує, так як абсолютна ступінь надійності не може бути теоретично доведена і, отже, є недосяжною. Однак важливо знати, наскільки надійно конкретне ПЗ[3], практика розробки якого передбачає пріоритет завдання забезпечення надійності над завданням його оцінки.

Метою даної роботи є встановлення наявності або відсутності залежності точності оцінювання надійності від мови програмування для найбільш вживаних моделі надійності. Для досягнення цієї мети в роботі розв'язано такі завдання:

- виконано моделювання надійності для декількох програмних систем за допомогою основних (базових) моделей надійності програмного забезпечення;
- досліджено точність моделювання в залежності від мови програмування і моделі.

1. Класифікація моделей надійності

На сьогодні створено декілька (за різними оцінками – близько п'ятнадцяти [4]) класифікаційних схем. Але найбільш поширена класифікація запропонована в роботі [2]. В ній виділяються такі класифікаційні ознаки:

1. Час моделі. Поточний астрономічний час (календарний час) або процесорний час, витрачений на роботу з цією програмною системою (ПС) до моменту виявлення дефекту. Процесорний час є дискретним і складається з окремих інтервалів, впродовж яких ця ПС знаходиться в активному стані. Процесорний час використовується в моделі розподілу Вейбула [5] та моделі Муси [6]). Астрономічний час використовується в більшості моделей (модель Джелінського-Моранди [7], модель нерівномірного Пуасонівського процесу [8], модель Шнайдевінда[9], модель Муси-Окумото [10], модель Дюена[11], геометрична модель Моранди [12] та *S*-подібна модель [13]). Це обумовлено тим, що астрономі-

чний час є найбільш зручним обліком часу для людини.

2. Категорія моделі. Категорію обумовлює первинна кількість дефектів в програмній системі. За цією ознакою моделі діляться на кінцеві (моделі Джелінського-Моранди, нерівномірного Пуасонівського процесу, Шнайдевінда, Муси, розподілу Вейбула та *S*-подібна модель) і нескінченні (моделі Дюена, Муси-Окумото та геометрична модель Моранди).

3. Тип моделі. Визначає розподіл ймовірності настання випадкової події, під якою мається на увазі виявлення дефекту. Розподіл Пуассона моделює випадкову величину, що є числом подій, що сталися за фіксований час, за умови, що ці події відбуваються з деякою фіксованою середньою інтенсивністю і незалежно один від одного (моделі нерівномірного Пуасонівського процесу, Шнайдевінда, Муси, Дюена, Муси-Окумото, *S*-подібна модель та геометрична модель Моранди). Біноміальний розподіл (розподіл Бернуллі) визначає розподіл ймовірності кількості поява деякої події при повторних незалежних випробуваннях, якщо ймовірність появи цієї події в кожному з випробувань постійна (моделі Джелінського-Моранди та розподілу Вейбула,).

4. Клас моделі (тільки для категорії кінцевих моделей) та сімейство моделі **(**тільки для нескінченних моделей**)** визначає вид функції, що описує закон зміни інтенсивності прояву дефектів: експоненційний (моделі Джелінського-Моранди, нерівномірного Пуасонівського процесу, Шнайдевінда, Муси, Муси-Окумото, розподілу Вейбула та геометрична модель Моранди) і гамма-розподіл (*S*-подібна модель та модель Дюена).

Тут слід відзначити, що усі перелічені моделі базуються на теорії ймовірності, яка має справу лише із зовнішнім проявом якого-небудь процесу, але не враховує внутрішніх механізмів цього процесу.

2. Проблема вибору моделей надійності

Велика кількість моделей змушує підняти питання вибору однієї оптимальної, що дозволить максимально точно проаналізувати надійність конкретного програмного продукту та спрогнозувати його відмови. Вибір оптимальної МНПЗ для використання в конкретному випадку представляє великий інтерес для дослідників в галузі надійності ПЗ.

Практично кожна компанія, яка розробляє ПЗ має вирішити свою проблему вибору моделі, враховуючи досвід і навички своїх програмістів і дизайнерів, специфіку розроблюваних систем, вимоги до надійності і безпеки, економічні фактори і т. д. [14, 15, 16].

Оцінка моделей зростання надійності ПЗ за своєю суттю складна через відсутність об'єктивних заходів. Оскільки вона може включати безліч критеріїв вибору, проблема вибору може бути представлена, як проблема прийняття рішення по безлічі критеріїв (ПРБК). ПРБК-підхід для ранжирування вибору різних МНПЗ в області надійності ПЗ з використанням методу на основі ентропійної відстані (EDBA), шістнадцяти МНПЗ і семи критеріїв вибору в наборі даних про збої в реальному часі, який представлений у роботі [17]. Але слід зауважити, що метод не оцінює критичності різних критеріїв відбору, що беруть участь в процесі оцінювання, та не дозволяє враховувати велику кількість критеріїв.

Метод вибору оптимальної МНПЗ може виконуватися за допомогою глибокого навчання і порівняння методів для оцінки сукупного числа виявлених несправностей, з використанням даних про несправності реальних програмних проектів, що описується у роботі [18], Зокрема, робиться зауваження, що важко оцінити надійність, використовуючи лише дані попередніх несправностей, оскільки результати оцінки на основі даних про минулі помилки не можуть бути гарантованими для майбутніх наборів даних фактичних програмних проектів. .

МНПЗ можна класифікувати відповідно до фаз життєвого циклу розробки програмного забезпечення, визначено ряд критеріїв (за рівнем важливості) для вибору моделі надійності програмного забезпечення і запропонований алгоритм для вибору МНПЗ на основі цих критеріїв, який може використовуватися на різних етапах життєвого циклу розробки ПЗ і застосовуватися до різних класів МНПЗ, як у роботі [19].

В роботі [20] запропоновано розв'язання проблеми вибору МПЗ за допомогою нечіткої нейронної мережі з двома змінними, яка дозволяє вирішувати проблему порівняння і вибору МНПЗ. Автори стверджують, що даний метод дозволяє ефективно підібрати модель надійності за відсутності великої кількості даних, на початковому етапі розробки програмної системи. У роботі не зазначено чи ефективним буде розроблене рішення на більш фінальних етапах розробки ПО за наявність великої кількості змінних да даних о системі.

Детерміновану кількісну модель, засновану на методі дистанційного підходу для оцінки, оптимального вибору та ранжування МНПЗ на основі евклідової складеної відстані кожної альтернативи розроблено [21]. Запропонований метод підходить для ранжирування МНПЗ, заснованих на ряді суперечливих критеріїв, взятих разом, і для проведення тестів аналізу чутливості, щоб визначити їх найменш домінуючі критерії. Очевидно, що за відсутності саме цих специфічних даних ефективність запропонованого методу береться під сумнів.

Однак інструменти і методи для вибору моделі надійності програмного забезпечення, які описано в літературі, не можна використовувати з високим ступенем достовірності, оскільки вони використовують обмежену кількість критеріїв вибору моделі. Відсутність систематизованості процесу та моделей вибору МНПЗ для аналізу реального програмного продукту може значно ускладнити розрахунок надійності та поставити під сумнів його ефективність.

3. Дослідження моделей надійності

У процесі дослідження точності оцінювання надійності програмного забезпечення розглядаються 9 найбільш поширених моделей надійності програмного забезпечення:

1. модель Джелінського-Моранди;
2. модель нерівномірного Пуасонівського процесу (Гела – Окумото);
3. модель Шнайдевінда;
4. модель Муси;
5. модель Вейбула;
6. *S*-подібна модель;
7. модель Дюена;
8. геометрична модель Моранди;
9. модель Муси-Окумото.

Кожна з цих моделей є типовим представником свого відповідного класу в системі класифікації. Саме на базі цих моделей в рамках відповідного класу, розроблено велику кількість модифікацій, які й обумовлюють вказану раніш велику кількість моделей. Використання для дослідження таких класоутворюючих моделей дозволяє зменшити кількість розрахунків з одночасним збереженням широти охоплення всіх існуючих моделей.

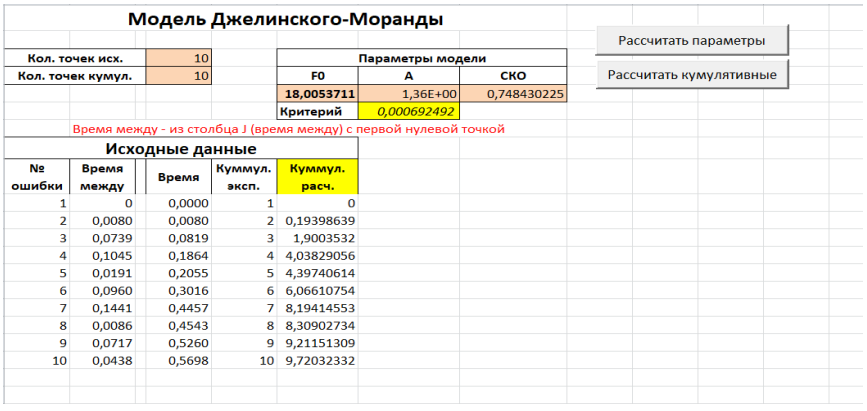
Для дослідження обрано часові ряди виявлення дефектів у 40 програмних продуктах [22], створених за допомогою різних мов програмування: JavaScript, Ruby, Python, Objective-C, C ++, Scala, C #, PHP, C, Java, Rust, ActionScript. Кількість досліджених програмних продуктів в залежності від мови програмування приведена в табл.1

Таблиця 1

Кількість досліджених програмних продуктів в залежності від мови програмування

Мова програмування	Кількість рядів
JavaScript	11
Ruby	6
Python	2
Objective-C	8
C++	1
Scala	1
C#	1
PHP	2
C	5
Java	1
Rust	1
ActionScript	1

Для аналізу обраних часових рядів за допомогою моделей надійності створене відповідне програмне забезпечення в середовищі Microsoft Excel. На рис.1 наведено приклад розрахунку за моделлю Джелінського-Моранди.



Модель Джелинского-Моранды

Рассчитать параметры

Рассчитать кумулятивные

Кол. точек исх.	10
Кол. точек кумул.	10

Параметры модели		
F0	A	СКО
18,0053711	1,36E+00	0,748430225
Критерий	0,000692492	

Время между - из столбца J (время между) с первой нулевой точкой

Исходные данные

№ ошибки	Время между	Время	Куммул. эксп.	Куммул. расч.
1	0	0,0000	1	0
2	0,0080	0,0080	2	0,19398639
3	0,0739	0,0819	3	1,9003532
4	0,1045	0,1864	4	4,03829056
5	0,0191	0,2055	5	4,39740614
6	0,0960	0,3016	6	6,06610754
7	0,1441	0,4457	7	8,19414553
8	0,0086	0,4543	8	8,30902734
9	0,0717	0,5260	9	9,21151309
10	0,0438	0,5698	10	9,72032332

Рис. 1. Ілюстрація розрахунку показників надійності за моделлю Джелінського-Моранди

Метою розрахунків було встановлення точності оцінювання надійності кожною моделлю для кожної програмної системи. Точність визначалася, як середньоквадратичне відхилення (СКВ) змодельованої та реальної кумулятивної кривої виявлення дефектів.

СКВ, згідно із [23], визначається за формулою

$$СКВ = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(x_c - x_o\right)^2},$$

де x_c – розхована за моделлю кількість дефектів, x_o – реальна кількість дефектів, яка виявлена в експерименті, N – кількість спостережень.

Ця характеристика дозволяє оцінити, наскільки відповідна модель може бути застосована для оцінки надійності програмного продукту, що розробляється за допомогою певної мови програмування.

На основі отриманих результатів сформовано зведену таблицю, у якій показано діапазон СКВ для кожної моделі надійності та кожної мови програмування (табл. 2).

Таблиця 2

Середнє квадратичне відхилення за різними моделями

Мова	Кількість інтервалів	Джелінського-Моранди			Гела-Окумото (NHPP)			Шнайдевінда		
		Середнє	Мін.	Макс.	Середнє	Мін.	Макс.	Середнє	Мін.	Макс.
JavaScript	112	1175,42	0,29	24046,18	11,83	0,45	343,02	3,58	0,18	36,69
Ruby	64	5785,91	0,23	112757,52	1,79	0,37	10,33	4,88	0,15	25,23
Python	30	0,71	0,34	1,90	1,72	0,39	7,08	1,38	0,09	5,28
Objective-C	75	1582,39	0,23	23730,59	3,91	0,40	25,94	38,83	0,05	306,79
C++	4	1,24	1,24	1,24	1,98	1,23	3,83	-	0,00	0,00
Scala	5	157,85	36,97	371,96	1,41	0,76	2,11	93,65	40,57	146,72
C#	4	0,99	0,99	0,99	15,12	0,52	45,75	0,31	0,10	0,58
PHP	15	37,90	1,30	123,53	2,37	0,61	9,35	25,21	13,00	37,80
C	29	186049,05	0,36	2774008,03	1,57	0,31	3,97	15,32	0,23	73,46
Java	11	2,77	0,53	6,40	20,11	0,50	94,15	1,03	0,28	3,25
Rust	10	0,92	0,37	1,62	1,78	0,50	4,43	0,37	0,08	0,63
ActionScript	15	0,77	0,28	2,13	2,16	0,35	15,60	0,28	0,04	0,69

Таблиця 2. Продовження

Мова	Кількість інтервалів	Муса			Розподілення Вейбулла			S-подібна		
		Середнє	Мін.	Макс.	Середнє	Мін.	Макс.	Середнє	Мін.	Макс.
JavaScript	112	2,31	0,04	28,48	3,29	0,07	166,48	12,70	0,03	391,75
Ruby	64	0,94	0,09	6,92	0,84	0,08	4,28	3,90	0,08	16,73
Python	30	0,49	0,19	1,60	0,94	0,05	6,74	2,65	0,08	15,14
Objective-C	75	0,90	0,11	7,03	1,24	0,01	11,30	6,34	0,05	136,74
C++	4	1,43	1,43	1,43	0,41	0,27	0,65	2,70	1,65	4,08
Scala	5	0,78	0,39	1,39	0,43	0,17	0,85	2,24	0,11	9,39
C#	4	13,54	0,72	26,37	3,28	0,31	10,05	1,87	0,72	3,61
PHP	15	1,33	0,57	2,52	0,57	0,11	1,36	3,50	0,27	12,52
C	29	0,83	0,17	2,35	0,71	0,06	3,84	2,64	0,09	8,47
Java	11	8,97	0,27	25,45	7,95	0,29	27,78	13,79	0,18	59,10
Rust	10	16,65	0,23	113,53	0,44	0,07	1,08	5,38	0,56	33,79
ActionScript	15	1,77	0,16	6,78	0,64	0,11	1,78	1,36	0,12	5,04

Таблиця 2. Продовження

Мова	Кількість інтервалів	Дюена			Геометрична Моранди			Логарифмічна Муси-Окумото		
		Середнє	Мін.	Макс.	Середнє	Мін.	Макс.	Середнє	Мін.	Макс.
JavaScript	112	1,84	0,13	32,61	3,03	0,14	44,91	34,76	0,28	303,36
Ruby	64	0,89	0,11	3,22	2,38	0,17	12,97	6,00	0,25	100,17
Python	30	0,77	0,15	3,97	2,27	0,49	6,09	0,88	0,37	2,17
Objective-C	75	1,24	0,17	9,34	2,44	0,31	15,81	0,97	0,24	4,19
C++	4	0,37	0,18	0,79	2,20	1,89	2,46	0,43	0,30	0,68
Scala	5	0,45	0,18	0,73	1,67	0,14	2,62	0,30	0,30	0,30
C#	4	2,31	0,99	4,08	4,11	0,39	10,72			
PHP	15	0,53	0,18	1,13	2,35	0,14	4,85	3,14	0,38	10,51
C	29	0,69	0,24	1,28	2,37	0,66	7,02	0,54	0,37	1,02
Java	11	3,84	0,38	11,96	5,15	1,79	16,91	313,89	313,89	313,89
Rust	10	0,90	0,34	1,51	1,76	0,49	3,69	0,46	0,29	0,62
ActionScript	15	0,87	0,22	2,13	1,90	0,35	3,50	0,43	0,34	0,56

Результати, представлені в таблиці 2, демонструють неоднорідність ефективності певної моделі при її використанні для розрахунку надійності ПЗ, написаного на певній мові програмування. Жодну з моделей не можна використовувати для усіх МП.

Середні значення СКВ при використанні моделей Гели-Окумото, Муси, розподілення Вейбула, Дюена та геометричної моделі Моранди для усіх досліджених мов програмування не перевищують 20,1.

Цікавим є питання – а чи є якась модель надійності найкращою для певної мови програмування? Для відповіді на це питання було проведено кореляційний аналіз між мовами програмування та моделями надійності. В якості даних для розрахунку коефіцієнту кореляції взято середні значення СКВ із таблиці 2. Зважаючи на те, що одна з характеристик, а саме мова програмування, не є числовим значенням, кожній мові було поставлено у відповідність порядковий номер. Дані для розрахунку кореляції наведено в таблиці 3.

Таблиця 3

Дані для розрахунку кореляції між мовою програмування та точністю оцінки за моделями надійності

Мова	№	Дж-Мор.	Гела-Окум	Шнайд	Муси	Вейб.	S-под.	Дюена	Геом. Мор.	Лог. Муси
JavaScript	1	1175,42	11,83	3,58	2,31	3,29	12,7	1,84	3,03	34,76
Ruby	2	5785,91	1,79	4,88	0,94	0,84	3,9	0,89	2,38	6
Python	3	0,71	1,72	1,38	0,49	0,94	2,65	0,77	2,27	0,88
Objective-C	4	1582,39	3,91	38,83	0,9	1,24	6,34	1,24	2,44	0,97
C++	5	1,24	1,98		1,43	0,41	2,7	0,37	2,2	0,43
Scala	6	157,85	1,41	93,65	0,78	0,43	2,24	0,45	1,67	0,3
C#	7	0,99	15,12	0,31	13,54	3,28	1,87	2,31	4,11	
PHP	8	37,9	2,37	25,21	1,33	0,57	3,5	0,53	2,35	3,14
C	9	186049	1,57	15,32	0,83	0,71	2,64	0,69	2,37	0,54
Java	10	2,77	20,11	1,03	8,97	7,95	13,79	3,84	5,15	313,89
Rust	11	0,92	1,78	0,37	16,65	0,44	5,38	0,9	1,76	0,46
ActionScript	12	0,77	2,16	0,28	1,77	0,64	1,36	0,87	1,9	0,43

Зважаючи на те, що номер мови програмування є тільки її рангом, а не фізично пов'язаною із нею величиною, для розрахунків використано коефіцієнт рангової кореляції Спірмена [24]. Результати розрахунків кореляції між мовою програмування та точністю оцінок за моделями надійності наведено в таблиці 4.

Таблиця 4.

Коефіцієнти кореляції

Мова програмування	Коефіцієнт Спірмена
JavaScript	-0,5378
Ruby	-0,5475
Python	0,19822
Objective-C	-0,5561
C++	-0,1841
Scala	-0,6488
C#	-0,2411
PHP	-0,6315
C	-0,5477
Java	0,54067
Rust	-0,1046
ActionScript	-0,0911

Від'ємне значення коефіцієнту кореляції, як відомо, відповідає випадку, коли при зростанні показника за віссю X (номер мови), показник за віссю Y (точність оцінювання) зменшується. Зважаючи на те, що номери мов програмування було проставлено довільно, ми можемо використовувати модуль коефіцієнта кореляції.

Згідно із шкалою Чеддока [24], високим ступенем зв'язку між величинами відповідають коефіцієнти кореляції, вищі ніж 0,7. Тому на підставі даних таблиці 4 можна зробити висновок, що зв'язок між точністю оцінок надійності програмного забезпечення та мовою програмування є несуттєвим, або зовсім відсутній.

Висновки

Аналізуючи припущення та дані, отримані в процесі моделювання можна зробити наступні висновки:

1. Кожна з моделей моделі надійності програмного забезпечення показує різні за точністю результати розрахунку надійності розглянутих програмних систем.
2. З отриманих даних видна відсутність залежності точності оцінювання від конкретної моделі та конкретної мови програмування. Модель, яка може бути ефективною для розрахунку надійності ПЗ, яке було написане на одній мові програмування, буде не ефективною для розрахунку надійності ПЗ, написаного на іншій мові програмування.
3. Виявлено, що для деяких рядів даних певні моделі не є працездатними. Це стосується особливо рядів, кумулятивна крива для яких має експоненційне зростання у часі.
4. На теперішний час не існує однієї універсальної моделі, яка пропонувала би однаково точний розрахунок ефективності програмного забезпечення, написаного на усіх мовах програмування.

Однак, для підвищення достовірності цих висновків, слід збільшити кількість програмних продуктів для аналізу. Це можна вважати завданням подальших досліджень.

Список використаної літератури

1. Маевский, Д. А., Яремчук, С. А. Сравнительный анализ моделей надежности программного обеспечения на этапе эксплуатации [Текст] – Праці Одеського політехнічного університету. –2011. Вип. 1(35).

2. Lyu, M. R. Handbook of Software Reliability Engineering [Text] / M. R. Lyu. – New York: McGraw-Hill Company. – 1996. – 805 p.

3. Василенко, Н. В., Макаров, В. А. Модели оценки надежности программного обеспечения [Текст]. –2004.– Вестник новгородского государственного университета № 28.

4. Харченко, В. С. Методы моделирования и оценки качества и надежности программного обеспечения [Текст] / В. С. Харченко, В. В. Скляр, О. М. Тарасюк. Учеб. пособие. – Харьков: Нац. аэрокосм. ун-т ХАИ. –2004. – 159 с.

5. Nadarajah, S., & Kotz, S. On Some Recent Modifications of Weibull Distribution. [Text] IEEE Transactions on Reliability, –2005. – 54(4), 561–562. doi:10.1109/tr.2005.858811

6. Yin, L., Zhi-An, S., & Jiang-Ting-Ting. A Software Reliability Test Suite Generating Approach Based on Hybrid Model for Complex Equipment System. [Text]. –2017. – International Conference on Dependable Systems and Their Applications (DSA). doi:10.1109/dsa.2017.32

7. Luo, Z., Cao, P., Tang, G., & Wu, L. A Modification to the Jelinski-Moranda Software Reliability Growth Model Based on Cloud Model Theory. [Text] . –2011. –Seventh International Conference on Computational Intelligence and Security. doi:10.1109/cis.2011.51

8. Keiller, P. A., & Mazzuchi, T. A. (n.d.). Enhancing the predictive performance of the Goel-Okumoto software reliability growth model. Annual Reliability and Maintainability Symposium. [Text]. –2000. – Proceedings. International Symposium on Product Quality and Integrity. doi:10.1109/rams.2000.816292

9. Liu, Q., Wu, J., & Tian, J. Updated Schneidewind model with single change-point by geometrical method. [Text]. –2012. – Proceedings of 2012 2nd International Conference on Computer Science and Network Technology. doi:10.1109/ICCSNT.2012.6526166

10. Ullah, N., Morisio, M., & Vetro, A. A Comparative Analysis of Software Reliability Growth Models using Defects Data of Closed and Open Source Software [Text]. –2012. – 35th Annual IEEE Software Engineering. doi:10.1109/sew.2012.26

11. Cai, Z., Chen, Y., Zhang, Z., & Wang, L. Method on integrated reliability assessment of test data based on Duane model [Text]. – 2014. – 10th International Conference on Reliability, Maintainability and Safety (ICRMS) doi:10.1109/icrms.2014.7107295

12. Boland, P. J., & Singh, H. A birth-process approach to Moranda's geometric software-reliability model [Text]. –2003.– IEEE Transactions on Reliability, 52(2), 168–174. doi:10.1109/tr.2003.813166

13. Kashyap, E., & Rana, A. A Comparative Study of S-shape and Concave Software Reliability Growth Models [Text]. –2015. – International Conference on Computational Intelligence and Communication Networks (CICN) doi:10.1109/cicn.2015.280

14. Kharchenko, V. S., Tarasyuk, O. M., Sklyar, V. V., & Dubnitsky, V. Y. (n.d.). The method of software reliability growth models choice using assumptions matrix [Text]. –2002. – Proceedings 26th Annual International Computer Software and Applications. doi:10.1109/cmpsac.2002.1045062

15. Garg, M., & Lai, R. Software reliability model selection for component-based real-time systems [Text]. – 2014. – International Conference on Data and Software Engineering (ICODSE). doi:10.1109/icodse.2014.7062706

16. Krini, J., & Borcsok, J. Basic concept for the selection of an appropriate "software failure prediction model." [Text]. – 2015. – Third World Conference on Complex Systems (WCCS). doi:10.1109/icocs.2015.7483233

17. Gupta, A., Gupta, N., Garg, R., & Kumar, R. Evaluation, Selection and Ranking of Software Reliability Growth Models using Multi criteria Decision making Approach [Text]. – 2018. – 4th International Conference on Computing Communication and Automation (ICCCA). doi:10.1109/ccaa.2018.8777644

18. Tamura, Y., Matsumoto, M., & Yamada, S. Software Reliability Model Selection Based on Deep Learning [Text]. – 2016. – International Conference on Industrial Engineering, Management Science and Application (ICIMSA). doi:10.1109/icimsa.2016.7504034

19. Asad, C. A., Ullah, M. I., & Rehman, M. J. (n.d.). An approach for software reliability model selection [Text]. –2004. – Proceedings of the 28th Annual International Computer Software and Applications Conference, COMPSAC 2004. doi:10.1109/cmpsac.2004.1342891

20. Chao, B., Xu, R., & Gu, M. A study of software reliability models choice based on bidirectional learning fuzzy neural network [Text]. – 2009. – IEEE International Conference on Industrial Engineering and Engineering Management. doi:10.1109/ieem.2009.5373363

21. Sharma, K., Garg, R., Nagpal, C. K., & Garg, R. K. Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach. [Text]. –2010. – IEEE Transactions on Reliability, 59(2), 266–276. doi:10.1109/tr.2010.2048657

22. Open-source version control system [Электронный ресурс]. – Режим доступа: https://github.com/

23. Poli, Attilio & Cirillo, Mario. On the use of the normalized mean square error in evaluating dispersion model performance. Atmospheric Environment. Part A. General Topics. [Text]. –1993. – 27. 2427–2434. 10.1016/0960-1686(93)90410-Z.

24. James, G. and Witten, D. An Introduction to Statistical Learning: With Applications In R. New York [etc.]: Springer. [Text]. –2013.

References

1. Maevsky, D. A., Yaremchuk, S. A. (2011) "Comparative analysis of software reliability models during operation" [Sravnitelnyj analiz modelej nadezhnosti programmnogo obespecheniya na etape ekspluatacii] Proceedings of Odessa Polytechnic University, iss. 1.35.

2. Lyu, M., (1996). Handbook of Software Reliability Engineering. New York: McGraw-Hill Company, p.805.

3. Vasilenko, N. V., Makarov, V. A. (2004) "Software Reliability Assessment Models" [Modeli ocenki nadezhnosti programmnogo obespecheniya] Bulletin of Novgorod State University № 28

4. Harchenko, V. S., Sklyar, V. V., Tarasyuk, O. M. (2004)"Methods for modeling and evaluating the quality and reliability of software" [Metody modelirovaniya i ocenki kachestva i nadezhnosti programmnogo obespecheniya] Kharkiv National aerospace un-t KhAI, p. 159.

5. Nadarajah, S., Kotz, S. (2005). "On Some Recent Modifications of Weibull Distribution" IEEE Transactions on Reliability, 54(4), 561–562. doi:10.1109/tr.2005.858811

6. Yin, L., Zhi-An, S., & Jiang-Ting-Ting. (2017). A Software Reliability Test Suite Generating Approach Based on Hybrid Model for Complex Equipment System. 2017 International Conference on Dependable Systems and Their Applications (DSA). doi:10.1109/dsa.2017.32

7. Luo, Z., Cao, P., Tang, G., & Wu, L. (2011). A Modification to the Jelinski-Moranda Software Reliability Growth Model Based on Cloud Model Theory. 2011 Seventh International Conference on Computational Intelligence and Security. doi:10.1109/cis.2011.51

8. Keiller, P. A., & Mazzuchi, T. A. (n.d.). (2000) Enhancing the predictive performance of the Goel-Okumoto software reliability growth model. Annual Reliability and Maintainability Symposium. 2000 Proceedings. International Symposium on Product Quality and Integrity. doi:10.1109/rams.2000.816292

9. Liu, Q., Wu, J., & Tian, J. (2012). Updated Schneidewind model with single change-point by geometrical method. Proceedings of 2012 2nd International Conference on Computer Science and Network Technology. doi:10.1109/ICCSNT.2012.6526166

10. Ullah, N., Morisio, M., & Vetro, A. (2012). A Comparative Analysis of Software Reliability Growth Models using Defects Data of Closed and Open Source Software. 2012 35th Annual IEEE Software Engineering. doi:10.1109/sew.2012.26

11. Cai, Z., Chen, Y., Zhang, Z., & Wang, L. (2014). Method on integrated reliability assessment of test data based on Duane model. 2014 10th International Conference on Reliability, Maintainability and Safety (ICRMS) doi:10.1109/icrms.2014.7107295

12. Boland, P. J., & Singh, H. (2003). A birth-process approach to Moranda's geometric software-reliability model. IEEE Transactions on Reliability, 52(2), 168–174. doi:10.1109/tr.2003.813166

13. Kashyap, E., & Rana, A. (2015). A Comparative Study of S-shape and Concave Software Reliability Growth Models. 2015 International Conference on Computational Intelligence and Communication Networks (CICN) doi:10.1109/cicn.2015.280

14. Kharchenko, V. S., Tarasyuk, O. M., Sklyar, V. V., & Dubnitsky, V. Y. (n.d.). (2002) The method of software reliability growth models choice

using assumptions matrix. Proceedings 26th Annual International Computer Software and Applications. doi:10.1109/cmpsac.2002.1045062

15. Garg, M., & Lai, R. (2014). Software reliability model selection for component-based real-time systems. 2014 International Conference on Data and Software Engineering (ICODSE). doi:10.1109/icodse.2014.7062706

16. Krini, J., & Borcsok, J. (2015). Basic concept for the selection of an appropriate "software failure prediction model." 2015 Third World Conference on Complex Systems (WCCS). doi:10.1109/icocs.2015.7483233

17. Gupta, A., Gupta, N., Garg, R., & Kumar, R. (2018). Evaluation, Selection and Ranking of Software Reliability Growth Models using Multi criteria Decision making Approach. 2018 4th International Conference on Computing Communication and Automation (ICCCA). doi:10.1109/ccaa.2018.8777644

18. Tamura, Y., Matsumoto, M., & Yamada, S. (2016). Software Reliability Model Selection Based on Deep Learning. 2016 International Conference on Industrial Engineering, Management Science and Application (ICIMSA). doi:10.1109/icimsa.2016.7504034

19. Asad, C. A., Ullah, M. I., & Rehman, M. J . (n.d.). (2004) An approach for software reliability model selection. Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004. doi:10.1109/cmpsac.2004.1342891

20. Chao, B., Xu, R., & Gu, M. (2009). A study of software reliability models choice based on bidirectional learning fuzzy neural network. 2009 IEEE International Conference on Industrial Engineering and Engineering Management. doi:10.1109/ieem.2009.5373363

21. Sharma, K., Garg, R., Nagpal, C. K., & Garg, R. K. (2010). Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach. IEEE Transactions on Reliability, 59(2), 266–276. doi:10.1109/tr.2010.2048657

22. Open-source version control system. – Access mode: https://github.com/

23. Poli, Attilio & Cirillo, Mario. (1993). On the use of the normalized mean square error in evaluating dispersion model performance. Atmospheric Environment. Part A. General Topics. 27. 2427–2434. 10.1016/0960-1686(93)90410-Z.

24. James, G. and Witten, D., (2013). An Introduction to Statistical Learning: With Applications In R. New York [etc.]: Springer.

RESEARCH OF ACCURACY OF SOFTWARE SOFTWARE RELIABILITY ASSESSMENT

D. A. Maevsky, O. V. Naidenko, E. J. Maevskaya, O. V. Strelzov, A. A. Naidenko
Odessa National Polytechnic University

Abstract. *The aim of the work is to establish the presence or absence of dependence of the accuracy of reliability assessment on the programming language and software reliability model. To this end, software reliability modeling was performed using the main reliability models, such as: Dzhelinsky-Moranda, non-uniform Poisson process (Gela-Okumoto), Schneide-Windows, Musa, Weibul model, S-Shaped model, Duena, geometric model of Moranda, Musa-Okumoto. The existence of the problem of choosing a reliability model, which is due to their large number, is noted. It is shown that the problem of choosing a model has not yet been resolved. For research, we selected time series for defect detection in 40 software systems written in various programming languages: JavaScript, Ruby, Python, Objective-C, C ++, Scala, C #, PHP, C, Java, Rust, ActionScript. The data source for the specified time series is the Internet resource Github.com. Modeling was carried out using specialized software developed by the authors. The simulation accuracy was estimated as the mean-squared deviation of the calculated cumulative defect detection curve from the real one. The dependence of the accuracy of software reliability assessment on the programming language and reliability model is given. Recommendations are given on choosing a model for a software system depending on the programming language. It is concluded that there is no one universal model that with acceptable accuracy would allow us to evaluate the reliability of a software system, regardless of the programming language in which it was written.*

Keywords: *estimation accuracy, software, reliability model, programming language, defect detection, failure.*

ИССЛЕДОВАНИЕ ТОЧНОСТИ ОЦЕНИВАНИЯ НАДЕЖНОСТИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Д. А. Маевский, Е. В. Найденко, Е. Ю. Маевская, О. В. Стрельцов, А. А. Найденко
Одесский национальный политехнический университет

Аннотация. *Приведена зависимость точности оценки надежности программного обеспечения от языка програмированния и модели надежности. Выполнено моделирование надежности для нескольких программных систем с помощью основных моделей. Показано, что точность оценивания надежности программного обеспечения не зависит от язіка программирования.*

Ключевые слова: *точность оценивания, программное обеспечение, модель надежности, язык программирования, выявление дефектов, отказ.*

Получено 09.12.19

Маєвський Дмитро Андрійович, Одеський національний політехнічний університет, доктор технічних наук, професор, завідувач кафедри електромеханічної інженерії. Просп. Шевченка, 1, Одесса, Україна, E-mail: Dmitry.A.Maevsky@gmail.com, тел. +38-048-705-84-54

Dmitry Maevsky, Odessa National Polytechnic University, Dr. of Science, Professor, Head of the Department of the electromechanical engineering, Shevchenko ave., 1, Odessa, Ukraine. E-mail: Dmitry.A.Maevsky@gmail.com, tel. +38-048-705-84-54

ORCID ID: 0000-0003-0666-6199

Найденко Олена Валеріївна, Одеський національний політехнічний університет, кандидат технічних наук, доцент, доцент кафедри електромеханічної інженерії Просп. Шевченка, 1, Одеса, Україна, E-mail: alena2808@ukr.net, тел. +38-048-705-84-67

Elena Naydenko, Odessa National Polytechnic University, Phd, Associate Professor, Department of the electromechanical engineering, Shevchenko ave., 1, Odessa, Ukraine. E-mail: alena2808@ukr.net, тел. +38-048-705-84-67

ORCID ID: 0000-0001-5684-5617

Маєвська Олена Юріївна, Одеський національний політехнічний університет, кандидат технічних наук, доцент, доцент кафедри електромеханічної інженерії Просп. Шевченка, 1, Одеса, Україна, E-mail: e.j.maevskaya@gmail.com тел. +38-048-705-84-85

Elena Maevskaya, Odessa National Polytechnic University, Phd, Associate Professor, Department of the electromechanical engineering, Shevchenko ave., 1, Odessa, Ukraine. E-mail: e.j.maevskaya@gmail.com тел. +38-048-705-84-85

ORCID ID: 0000-0001-6297-4255

Стрельцов Олег Васильович, Одеський національний політехнічний університет, кандидат технічних наук, доцент, доцент кафедри комп'ютерних систем, Просп. Шевченко, 1, Одеса, Україна, E-mail: streltsov.o.v@opu.ua тел. +38-048-705-84-73

Oleh Streltsov, Odessa National Polytechnic University, Phd, Associate Professor, Department of specialized computer systems, Shevchenko ave., 1, Odessa, Ukraine. E-mail: streltsov.o.v@opu.ua тел. +38-048-705-84-73

ORCID ID: 0000-0002-4691-5703

Найденко Артем Антонович, Одеський національний політехнічний університет, магістр кафедри комп"ютерних, Просп. Шевченка, 1, Одеса, Україна. E-mail: artemnaydenko@gmail.com тел. +38-050-103-37-35

Artem Naidenko, Odessa National Polytechnic University Master student Department of specialized computer systems, Shevchenko ave., 1, Odessa, Ukraine. E-mail: artemnaydenko@gmail.com тел. +38-050-103-37-35

ORCID ID: 0000-0002-5359-547X