

УДК 004.4'6

Шевченко В.Л., д.т.н, с.н.с.¹;Кіріпичников Ю.А. к.т.н.²;Федорієнко В.А.²;Головченко О.В.²¹ - Державний університет телекомунікацій;² - Центр воєнно-стратегічних досліджень Національного університету оборони України імені Івана Черняхівського

Модель оцінки надійності програмної компоненти єдиного інформаційного середовища

Модель оценки надежности
программной компоненты
единой информационной среды

The model for evaluating the
reliability of software components a
single information environment

Резюме. Стаття розкриває проблеми практичної оцінки надійності програмної складової єдиного інформаційного середовища. Запропоновано підхід для визначення необхідного рівня надійності програмної компоненти та визначення числа осіб відповідальних за працездатність програмного забезпечення в залежності від його складності.

Резюме. Статья раскрывает проблемы практической оценки надежности программной составляющей единой информационной среды. Предложен подход для определения необходимого уровня надежности программной компоненты и определение числа лиц ответственных за работоспособность программного обеспечения в зависимости от его сложности.

Resume. This article is devoted to the practice problems of assessing the reliability of software component a single information environment. In the article is proposed the approach to determine the required level of reliability of software components and to determine the number of persons responsible for the performance of software according to its complexity.

Ключові слова: надійність програмного забезпечення, метод Монте-Карло, єдине інформаційне середовище.

Ключевые слова: надежность программного обеспечения, метод Монте-Карло, единая информационная среда.

Keywords: software reliability, the Monte Carlo method, a single information environment.

Постановка проблеми. Робота будь-якого програмно-технічного комплексу може характеризуватися його *ефективністю* під якою розуміється сукупність властивостей, що забезпечують здатність виконання визначених завдань. Однією з таких властивостей є надійність комплексу. Взагалі, проблема надійності є ключовою по відношенню до інформаційних систем. Відмова у роботі (в тому числі – неправильне функціонування) інформаційних систем чи програмно-технічного комплексу) може призвести до значних наслідків. Тому, при розробці складних систем необхідно приділити увагу питанням надійності програмного забезпечення.

Сьогодні важливим питанням є розробка єдиного інформаційного середовища (ЄІС). Означене завдання було розглянуте в розрізі

адміністративно-господарської діяльності (АГД) для потреб військового відомства. У якості головної системи (ГС) ЄІС пропонуємо використовувати ERP-подібну систему [8]. Саме оцінка надійності ПЗ необхідна для вибору найкращого складу програмно-технічних засобів щодо створення єдиного інформаційного середовища. Зауважимо, що при оцінці надійності програмної компоненти ГС ЄІС важливо приділити увагу необхідній кількості спеціалістів (програмістів) для підтримки її в актуальному стані.

Ступінь розробленості проблеми. Поняття надійності інформаційних систем (програмно-технічного комплексу) розкрито у [1] та визначене в державних стандартах України [2, 3]. Питання оцінки надійності було

розглянуто в роботах авторів [1, 4]. Напрямок теорії надійності щодо оцінки програмного забезпечення використовує той же понятійний апарат, що і для технічних засобів. Підвищенню надійності ПЗ присвячена праця [5]. Але задача щодо моделювання надійності ПЗ за напрямом обслуговування програмної компоненти складних інформаційних систем військового призначення є новою.

Метою статті є аналіз підходів щодо оцінки надійності програмної складової єдиного інформаційного середовища для визначення числа осіб відповідальних за працездатність програмного забезпечення в залежності від його складності.

Виклад основного матеріалу. На початку проведемо уточнення понятійного апарату щодо надійності. Відповідно до ГОСТ 27.002-89, *надійність* ототожнюють з властивістю об'єкта зберігати певний термін у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах і умовах використання, технічного обслуговування, ремонтів, зберігання та транспортування [6].

Під *надійністю програмного забезпечення (ПЗ)* розуміється властивість ПЗ щодо виконання покладених функцій, зберігаючи свої характеристики у встановлених межах за певних умов експлуатації, що визначається його безвідмовністю і відновлюваністю [1].

Поява помилок у програмі характеризується відмовою – недопустимим відхиленням показників функціонування ПЗ від запропонованих вимог, а саме зниження значень за певними параметрами, що призводить до повного припинення виконання функцій програми або до короткочасного порушення ходу обробки інформації в системі [7].

Програмна компонента ГС ЄІС включає спеціальне програмне забезпечення (СПЗ) або “клієнтські” програми, які повинні бути надійними. Ці програми можуть виконуватися на графічному інтерфейсі користувача, але у більшості випадків – процедура виконання програми включає значну кількість кроків транзакції, що виконується на сервері (рівень застосування та бази даних).

У якості ГС ЄІС для АГД узяті ERP-система, ландшафт якої включає три системи :

– Система розробки – для проведення розробки спеціального програмного забезпечення (СПЗ) системи та налаштування системи.

– Система тестування – для перевірки якості роботи розроблених програм.

– Продуктивна система – для використання СПЗ на реальних даних без можливості зміни.

Цей системний ландшафт визначено в методологіях по веденню системних проєктів, наприклад, методологія Accelerated SAP, де розподіленість систем пояснюється високими вимогами до надійності ПЗ.

Усі роботи щодо розробки СПЗ виконуються в системі розробки. Для правильного подальшого супроводження розроблених програм проводиться їх тестування у відповідній системі. Розробка і тестування програм є взаємно поєднаними процедурами для підготовки якісного СПЗ, яке може бути “перенесене” до продуктивної системи. Підтримка в актуальному стані програмної компоненти може бути досягнута шляхом тестування та усунення помилок на стані підготовки СПЗ не лише в період розробки і перевірки якості програм, а й під час продуктивної експлуатації. Відповідно, частиною робіт щодо супроводження і підтримки ERP-системи у робочому стані є проведення постійних мікропроєктів по розробці та удосконаленню СПЗ у системі розробки, тестування на виявлення помилок та перенесення змін – відповідного СПЗ з уже виправленими помилками – до раніше запущеної продуктивної системи.

Для своєчасного виявлення та виправлення помилок повинен бути виділений штат фахівців із підтримки програмного забезпечення у робочому стані – програмісти. Необхідно розв’язати задачу щодо визначення необхідної кількості програмістів та рівень їхньої підготовленості для підтримки і супроводження ГС ЄІС на прикладі ERP-системи для АГД.

Для визначення надійності ПЗ беруть до уваги найпростіші потоки пуассонівських відмов, які мають такі властивості, як стаціонарність, ординарність та відсутність послідовності [1, 7]. Доведено [4], що число

помилку у програмі в кожен момент часу має розподіл Пуассона

$$P_m = \frac{a^m}{m!} e^{-a}, \quad (1)$$

де P_m – кількість помилок у програмі,

$m = 0, 1, 2, \dots$ – число подій,

a – параметр закону Пуассона.

Якщо випадкова величина у найпростішому потоці подій за час τ – деяке ціле позитивне значення m , то ця величина розподілена за законом Пуассона. Параметр цього розподілу змінюється після розподілу чергової помилки. Інтенсивність відмов $\lambda(\tau)$ вважається безперервною функцією, пропорційною числу помилок, які залишилися. З урахуванням введених параметрів і припущень очевидно, що

$$\lambda(\tau) = C \cdot m_0(\tau), \quad (2)$$

де $\lambda(\tau)$ – інтенсивність відмов,

C – коефіцієнт пропорційності, що враховує швидкодію “клієнтських” програм (СПЗ) і число команд у програмі,

$m_0(\tau)$ – початкове ціле значення випадкової величини за час τ .

Враховуючи випадковий характер подій і процесів, що виникають у системі для розрахунку показників надійності часто використовують прийоми і правила теорії ймовірностей і математичної статистики [4]. Наприклад, основними показниками безвідмовності, відповідно до ГОСТ 27.002, є ймовірність безвідмовної роботи, інтенсивність відмов і середній наробіток об'єктів до відмови [6]. Відповідно до Держстандарту України [3] рекомендовано вибирати такі види функцій розподілу для опису випадкових величин наробітку до відмови: експоненціальний розподіл, логарифмічно-нормальний розподіл, розподіл Вейбула, дифузійно-монотонний розподіл (DM-розподіл), дифузійно-немонотонний розподіл (DN-розподіл).

При визначенні характеристик надійності ПЗ (за умови його відновлення) будемо враховувати той факт, що помилки, які виникають при роботі програм усуваються, і їх кількість зменшується. Тому, їх інтенсивність знижується, а напрацювання програми на відмову збільшується. У зв'язку з такими припущеннями у літературі розглядається кілька моделей надійності ПЗ [1]: модель з дискретно-

понижувальною частотою появи помилок, модель з дискретним збільшенням напрацювання на відмову або помилку, експоненціальна модель надійності.

Як альтернатива зазначеним моделям в літературі зустрічаються моделі обрахунку надійності побудованих на одному з методів Монте-Карло. Цей метод відрізняється більшою простотою та кількістю доступних для обрахунку параметрів перспективної моделі і є гнучкішим при моделюванні оцінки надійності ПЗ за різних вхідних умов. Тому, в статті пропонується модель оцінки ПЗ, побудованій на методі Монте-Карло.

Взагалі, методи Монте-Карло – це загальна назва групи числових методів, оснований на одержанні великої кількості реалізацій стохастичного (випадкового) процесу, який формується у той спосіб, щоб його ймовірнісні характеристики збігалися з аналогічними величинами задачі, яку потрібно розв'язати.

Суть класичного методу Монте-Карло для визначення оцінки полягає в необхідності знаходження значення a деякої досліджуваної величини [9]. Слід зазначити, що метод Монте-Карло для визначення надійності ПЗ найкраще використовувати разом із ймовірнісними моделями, де набір даних проводиться по випадковим даними за правилами розподілу Пуассона [10]. Теорія цього методу вказує на те, як найбільш правильно підібрати випадкову величину, і знайти її можливі значення. Для цього, розробляються способи зменшення дисперсії випадкових величин, у результаті чого зменшується помилка. Однак цей метод є апроксимаційним, і не дозволяє отримати точне значення надійності.

Розглянемо початкові умови. Нехай, сервер обслуговує запити від N користувачів. У програмній компоненті ГС ЄІС виділяємо умовно область визначення вхідних даних, де рівномірно розташовані Er помилок.

Кожне k ($k = 1, 2, \dots, n$) “клієнтське” СПЗ створює пуассонівський потік даних до рівня бази даних (БД) сервера з інтенсивністю $\lambda_{обр}$.

Вхідними даними для оцінки надійності програмного забезпечення є:

P – кількість програмістів, які обслуговують систему;	s – коефіцієнт складності сервера;	M – кількість ітерацій;
K – кількість СПЗ (“клієнтських” програм);	$\lambda_{зв\pi\pi}$ – інтенсивність потоку звернень СПЗ до БД сервера за добу;	$K_{сп\pi}$ – кількість спроб для усереднення;
α – частка від загальної множини даних системи в запиті одного СПЗ ($0 < \alpha < 1$);	$\lambda_{вип\pi}$ – інтенсивність потоку виправлення помилки СПЗ одним програмістом за добу;	Er – початкова кількість помилок;
Δt – тривалість виконання кроку циклу при відладці СПЗ;	$\lambda_{внес}$ – інтенсивність внесення помилки при виправленні СПЗ одним програмістом за добу;	T – часовий проміжок узятий для тесту, діб.

Початкова кількість помилок визначається наступним чином: приймаємо ОВВД за одиницю; кожне СПЗ генерує запит α ; за час Δt СПЗ звертається до рівня БД сервера ($\Delta t \cdot \lambda_{зв\pi\pi}$) разів; за час Δt усі “клієнтські” програми звертаються до рівня БД сервера ($\Delta t \cdot \lambda_{зв\pi\pi} \cdot K$) разів. Обсяг даних, який буде використовуватися в ОВВД при цьому буде дорівнювати ($\Delta t \cdot \lambda_{зв\pi\pi} \cdot K \cdot \alpha$). Оскільки помилки розподілені рівномірно, то за час Δt буде визначено ($\Delta t \cdot \lambda_{пом}$). Тому,

$$\frac{\Delta t \cdot \lambda_{зв\pi\pi} \cdot K \cdot \alpha}{\Delta t \cdot \lambda_{пом}} = \frac{1}{Er}, \quad (3)$$

де $\lambda_{пом}$ – першочергова інтенсивність помилок у системі.

Звідки знаходимо Er :

$$Er = \frac{\lambda_{пом}}{\lambda_{зв\pi\pi} \cdot K \cdot \alpha}. \quad (4)$$

При цьому вважаємо, що кожна з “клієнтських” програм (СПЗ) K звернулася до сервера із запитом з даними, що не входять в область визначення вхідних даних (ОВВД). Частіше звернення від СПЗ до сервера відбувається шляхом надсилання однотипних запитів, тому вважаємо, що $K = 1$. Тоді оцінка першочергової кількості помилок матиме вид:

$$Er = \frac{\lambda_{пом}}{\lambda_{зв\pi\pi} \cdot \alpha}. \quad (5)$$

Якщо на інтервал часу Δt припадає більше однієї події ($\Delta t \cdot \lambda$), то інтенсивність становить менше одиниці [5].

Дані від СПЗ розподілені по області визначення даних за нормальним законом розподілу з характеристиками m_k і σ_k , де m_k має рівномірний розподіл між “клієнтськими” програмами (СПЗ) по всій області вхідних даних. На запит від клієнтського ПЗ сервер (з рівня БД) відсилає дані, які розподілені рівномірно по всій ОВВД (А, В). При надходженні запиту клієнтського ПЗ або відповіді сервера в область даних, що містить помилку, вважається, що

помилка виявлена і відповідний модуль виводиться з експлуатації для її усунення.

У випадку появи одиночної помилки при одному спостереженні виконуються наступні дії:

1. Визначається розміщення Er помилок “клієнтського” СПЗ на ОВВД;
2. Визначається розміщення $(s \cdot Er_k)/10$ помилок сервера на ОВВД, розподілених на ній рівномірно;
3. Для кожного з K “клієнтських” програм визначається на початку m_{ki} і σ_{ki} .
4. Далі за повтореннями, з кроком Δt для кожного СПЗ:

4.1. Якщо СПЗ справне, то воно може звертатися із запитом до БД сервера з інтенсивністю $\lambda_{зв\pi\pi}$. Ймовірність звернення “клієнтського” СПЗ до сервера становить $1 - e^{-\lambda_{зв\pi\pi} \cdot \Delta t}$. У разі звернення СПЗ до сервера визначається випадкова величина x_i , розподіл по нормальному закону з параметрами m_{ki} і σ_{ki} – вхідні дані для запиту сервера. Область, що містить вхідні дані запиту від однієї “клієнтської” програми до сервера на ОВВД, є випадковою величиною, розподіленою рівномірно на відрізьку ($\alpha/2$).

4.2. Якщо інтервал ($x_i \pm (\alpha/2)$) містить хоча б одну помилку на ОВВД, то вважають, що в СПЗ є помилка, і воно виводиться з експлуатації для її виправлення одним із вільних спеціалістом з обслуговування ПЗ (програмістом). Якщо вільних фахівців немає, то непрацездатне ПЗ переміщується у чергу і очікує, поки один із фахівців по працездатності ПЗ (програміст) буде вільним.

4.3. Якщо у запиті від СПЗ до сервера помилки немає, то цей запит направляється на сервер для виконання. При цьому визначається відповідь від сервера “клієнтському” ПЗ, за умови, що обсяг даних є випадковою величиною і запит розподілений рівномірно на відрізьку ($\alpha/10/2$). Якщо в область ($x_i \pm (\alpha/10/2)$) попадає хоча б одна помилка зі списку помилок

сервера, то вважають, що на сервері відбулася помилка. Це характеризується непрацездатним станом системи і проведенням робіт щодо усунення помилки з боку фахівців відповідальних за працездатність ПЗ (програмісти) на сервері зі швидкістю усунення $\lambda_{\text{випр}}$ кожен. Ймовірність виправлення помилки

одним фахівцем дорівнює $1 - e^{-\frac{\lambda_{\text{випр}} \cdot \Delta t}{s}}$.

4.4. Якщо в СПЗ є помилка і є вільний фахівець, відповідальний за працездатність ПЗ, то він виправляє помилку в СПЗ із ймовірністю $1 - e^{-\lambda_{\text{випр}} \Delta t}$.

4.5. Якщо помилка виправляється, то вона видаляється зі списку помилок СПЗ або сервера відповідно. Таким чином, ця помилка вже не виникає на інших “клієнтських” ПЗ. При виправленні помилки кожна особа відповідальна за працездатність ПЗ може внести нову помилку

з ймовірністю $1 - e^{-\lambda_{\text{внес}} \cdot \Delta t}$ або $p_{\text{внес}}$. Причому, якщо програміст вніс помилку в програму, то він може внести туди ще одну помилку з ймовірністю рівною квадрату ймовірності внесення попередньої помилки. Знову внесені помилки вносяться до списку помилок клієнтів або сервера відповідно. При цьому ці нові помилки не вважаються виявленими в СПЗ або сервері, тобто якщо виявлена помилка виправляється, то “клієнтське” ПЗ або сервер вважаються справними навіть, якщо при цьому були зроблені нові ідентичні помилки.

4.6. За один часовий проміжок Δt визначається сценарій обміну даними для усіх працюючих на цей момент часу СПЗ. Для несправних СПЗ або несправного сервера визначається ймовірність виправлення помилки.

5. У результаті визначається M повторень згідно з п. 4, і отримуємо одну реалізацію випадкових функцій $m_1^C(t)$, $m_2^C(t)$, і $\bar{p}(t)$ (середні чисельності працюючих, непрацюючих СПЗ та ймовірність роботи сервера) на тимчасовому інтервалі $M \cdot \Delta t$.

6. Випробування проводимо ще K разів, при цьому отримуємо K реалізацій випадкових функцій $m_1^C(t)$, $m_2^C(t)$, $m_2^{\bar{C}}(t)$ і $\bar{p}(t)$. Для кожного моменту часу t_j (для $j = 1, \dots, M$) з кроком Δt знаходимо статистичне середнє для цих функцій і отримуємо середні функції $\langle m_1^C(t) \rangle$, $\langle m_2^C(t) \rangle$, $\langle m_2^{\bar{C}}(t) \rangle$ і $\langle \bar{p}(t) \rangle$.

7. Також у процесі визначення надійності здійснюється: розрахунок поточного часу напрацювання до відмови; розрахунок середнього часу напрацювання до відмови за весь час визначення; розрахунок імовірності відмови ПЗ за одиницю часу як $P = (\langle \text{обсяг}$

$\text{запиту} \rangle \cdot \langle \text{кількість помилок в клієнтах і сервері} \rangle \cdot (\langle \text{кількість працюючих клієнтів} \rangle + 1) \cdot \langle \text{частота звернення} \rangle \cdot \langle \text{крок повторення за часом} \rangle$; кількість помилок на сервері дорівнює добутку кількості помилок СПЗ і коефіцієнту складності розділеного на 10.

Використання методу Монте-Карло дозволяє оцінити кількість помилок у програмі наступним чином – отримати з моделі розрахунковий результат, а потім шляхом визначення події непрацездатності ПЗ підібрати початкову кількість помилок. А також вирішити зворотну задачу, тобто, знаючи кількість осіб відповідальних за працездатність ПЗ, їх інтенсивність роботи та інтенсивність відмов на початку та в кінці дослідження можна визначити працевтрати стосовно виправлення оціночної кількості помилок.

Перейдемо відразу до вирішення зворотної задачі. Звернемо увагу на рівень підготовки спеціалістів, що спроможні підтримати програмне забезпечення. Програміст при виправленні помилки в програмі за деякий проміжок часу вносить помилку при виправленні машинного коду з певною інтенсивністю. У якості припущення прийемо показник інтенсивності внесення помилки при виправленні СПЗ ($\lambda_{\text{внес}}$), як міру кваліфікації спеціалістів. Будемо розрізняти програмістів за класністю:

- Спеціаліст I класу: $\lambda_{\text{внес}} = 0,1$.
- Спеціаліст II класу: $\lambda_{\text{внес}} = 0,3$.
- Спеціаліст III класу: $\lambda_{\text{внес}} = 0,7$.

Проведемо розрахунок необхідної кількості спеціалістів.

Вважаємо, що при створенні СПЗ у системі розробки усі “клієнтські” програми не є ідеальними. При їх перевірці якості у системі тестування виявляються недоліки – помилки, які необхідно виправити і потім знову відтестувати програму. Візьмемо дане твердження за правило при дослідженні моделі оцінки програмної компоненти ГС ЄІС.

Умови при проведенні розрахунку:

- Залучення до виправлення помилок програмістів на усьому часовому проміжку узятому для дослідження.
- Усереднення результатів дослідження за визначеною кількістю вимірювань.

В якості припущення прийемо коефіцієнт складності сервера ГС ЄІС відносно СПЗ: $s = 2$.

Спочатку розглянемо кваліфікацію спеціалістів по підтримці програмної компоненти ГС ЄІС, що відповідає I класу.

Вхідні дані при проведенні розрахунку

для I класу (розрахунок №1):

$P = 1, 2, \dots, n$, де $n = 10$;

$s = 2$;

$M = 50000$;

$K = 10$;

$\lambda_{\text{зв}} = 500$;

$K_{\text{спр}} = 50$;

$\alpha = 0,00001$;

$\lambda_{\text{випр}} = 1$;

$Er = 250$;

$\Delta t = 0,002$;

$\lambda_{\text{внес}} = 0,1$;

$T = 100$.

Почергово підставимо значення кількості програмістів (для зазначених вище умов): $P = 1, 2, \dots, n$, де $n = 10$.

Ступінчастий графік (рис. 1) відображає характер зміни кількості працездатних програм за одне вимірювання. Графік кривої на цьому ж рисунку відображає усереднене значення за 50 вимірювань. При застосуванні певних значень (зміну значення P) плавний характер кривої дасть більш чітке уявлення при моделюванні, тому в подальшому ми будемо використовувати саме середнє значення багаторазових вимірювань. Збільшивши штат програмістів до 2 і 3 чол. Кількість працюючих програм за увесь часовий проміжок буде зафіксовано на позначеннях 3 і 5, відповідно.

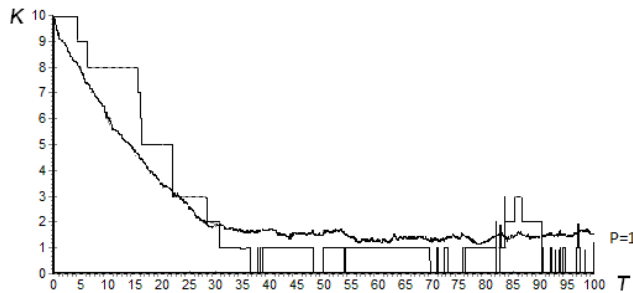


Рис. 1. Кількість працюючих програм при обслуговуванні 1 програмістом I класу (одиначна спроба і усереднена за 50 спроб)

На рис. 1 показано, що при одному програмісті на обслуговуванні системи ($P=1$) отримаємо стрімкий ріст помилок у програмах. Постійна поява помилок призводить до їх збільшення. Через це, з графіків – програмісти не спроможні відновити стійкий стан програмної компоненти системи, який був на початку періоду дослідження при своїй кількості за

штатом від 1 до 3 чол.

Розглянемо виправлення помилок при підтримці системи 4 програмістами (рис. 2). При 4 програмістах на обслуговуванні системи до кінця обраного періоду кількість робочих програм буде відповідати, як на початку періоду так і в кінці.

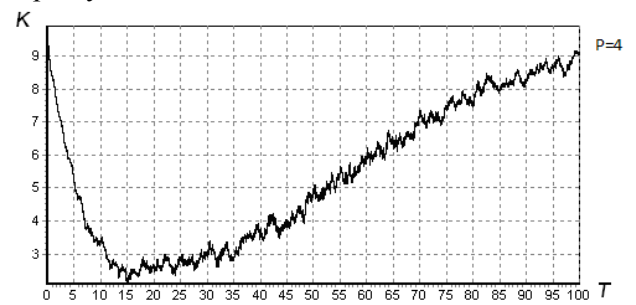


Рис. 2. Кількість працюючих програм системи при обслуговуванні 4 програмістами I класу

Спад V-подібної кривої (рис. 2). обумовлений тим, що на початку досліджуваного періоду усі програми містять в машинному коді помилки, які відразу ж виявляються з моменту експлуатації СПЗ. Кількість помилок за вихідними даними становить $Er = 250$. Спеціалісти при такому напливі помилок не спроможні їх миттєво вирішити. Крива має мінімальне значення працюючих програм на рівні цілого значення $K=2$ приблизно на 15 добу. Тому лише після значення $T=15$ діб починається ріст працюючих програм після усунення більшості помилок у СПЗ. Потім, іде стрімкий ріст на усьому часовому проміжку завдяки виявленню та усуненню помилок із заданою інтенсивністю. Крутість зростання залежить від кількості спеціалістів відповідальних за обслуговування програмної компоненти ГС ЄІС та їх кваліфікації (класності).

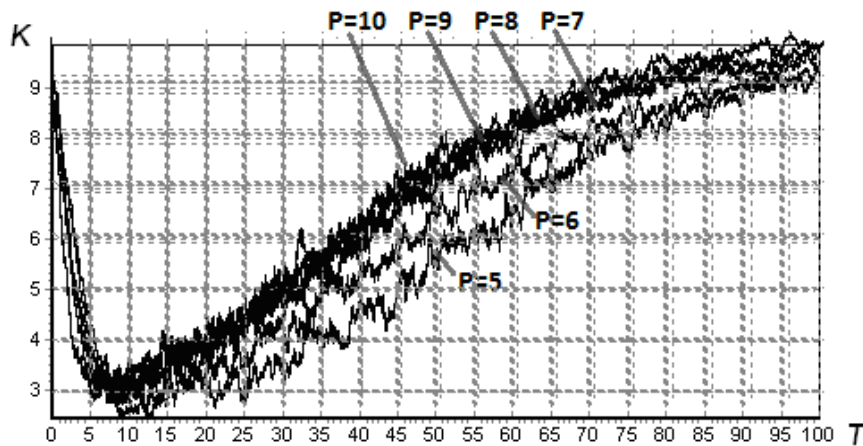


Рис. 3. Кількість працюючих програм системи при обслуговуванні 5-10 програмістами I класу

Поступово збільшимо число фахівців до 5-10 чол. (рис. 3). Очевидно, що при залученні більшої кількості програмістів відбувається своєчасне виправлення помилок у СПЗ системи. Але суміщення цих графіків дає наочну уяву про незначну розбіжність кількості працюючих програм за період у 100 діб. Різниця між кривими полягає у їх стрімкості досягнення вихідного значення числа працюючих програм K . Щільність графіків кривої для кількості програмістів $P=7, 8, 9$ і 10 чол., має мінімальну розбіжність, що свідчить про значний простій в роботі спеціалістів. А саме – після досягнення кривою початкового значення K (вісь ординат) за певний часовий проміжок T (вісь абсцис) спеціалісти будуть простоювати.

Порівнюючи рис. 2 і рис. 3 можна побачити схожість кривих. Відповідно, можна виділити певні вагові коефіцієнти (КРІ). КРІ1 – швидке досягнення значення працюючих програм (із розрахунку №1 КРІ1=1,2), чи КРІ2 – утримання більшої кількості фахівців. Використовуючи метод перебору, отримуємо шукане число спеціалістів із підтримки програмної компоненти (програмістів) I клас. Необхідна кількість: $P=4$ чол.

Змінимо у вхідних даних для $P=4$ показник інтенсивності внесення помилки при виправленні СПЗ ($\lambda_{\text{внес}}$), що відповідає за кваліфікацію фахівців. Нехай, $\lambda_{\text{внес}}=0,3$ – це спеціаліст II класу.

У результаті проведеного дослідження кількість штату з $P=4$ спеціалістів із середньою підготовкою (II клас) будуть неспроможні виправити помилки в СПЗ за встановлений термін. Методом перебору визначаємо, що найкращим чином цим умовам відповідає кількість програмістів: $P=7$ чол.

Змінивши у вхідних даних показник, що відповідає за кваліфікацію фахівців, $\lambda_{\text{внес}}=0,7$ – для спеціалісту III класу, можемо спостерігати

майже статичну поведінку кривої для різного числа спеціалістів III класу. Низький кваліфікаційний рівень осіб, відповідальних за працездатність ПЗ на ГС ЄІС, не спроможний підтримувати програмну компоненту у надійному стані. При виправленні помилки програмісти III класу породжують все нові і нові помилки, які остаточно виправити не вдається.

Таким чином, узагальнимо результати дослідження, отримані з моделі оцінки надійності ПЗ, побудованої за методом Монте-Карло. Якщо спеціалісти з підтримки програмної компоненти ГС ЄІС мають високий кваліфікаційний рівень (I клас), то відповідно вхідним даним розрахунку № 1, вони спроможні виконувати свої обов'язки у кількості 4 чол. Якщо спеціалісти мають середній кваліфікаційний рівень (II клас), то за вхідними даними розрахунку № 2 – у кількості 7 чол. При низькому кваліфікаційному рівні (III клас), за вхідними даними розрахунку № 3 – з поставленим завданням щодо обслуговування програмної компоненти, а саме проведенням своєчасного виправлення помилок у СПЗ навіть за чисельності штату в 20 чол. досягти за 100 днів значення 6 працюючих клієнтських програм із 10.

Висновок. Надійність ПЗ можливо завдяки його постійного обслуговування (виявлення та виправлення помилок в СПЗ) відповідних спеціалістів (програмістів). Завдяки використаній моделі оцінки ПЗ ми змогли спрогнозувати чисельність штату та кваліфікаційного рівня зазначених фахівців. У результаті стало очевидно, що при високому рівні програмісти спроможні виконати умовні завдання при нижчій чисельності, ніж при середньому рівні підготовки. При низькому кваліфікаційному рівні – програмісти із умовними завданнями дослідження не впоралися. Тому, спеціалістів відповідальних за

підтримку програмної складової ГС ЄІС необхідно готувати на відповідних курсах, а потім постійно проводити кваліфікаційне підтвердження (визначення класності) підготовки фахівців. У випадку наявності фахівців, кваліфікація яких відповідає вищому рівню, – економія на зменшенні чисельності штату можлива.

Подальші дослідження слід присвятити аналізу ефективності компонентів єдиного інформаційного середовища у розрізі АГД.

СПИСОК ЛІТЕРАТУРИ

1. Надёжность информационных систем: учебное пособие / Ю. Ю. Громов, О. Г. Иванова, Н. Г. Мосягина, К. А. Набатов. – Тамбов : ГОУ ВПО ТГТУ, 2010. – 160 с.
2. Надійність техніки. Терміни та визначення (ІЕС 50 (191)) : ДСТУ 2860-94. – [Чинний від 01.01.1996]. – К. : Держстандарт України, 1996. – 34 с. – (Національні стандарти України).
3. Надійність техніки. Методи розрахунку показників надійності: ДСТУ 2862-94. – [Чинний від 01.01.1996]. – К. : Держстандарт України, 1996. – 39 с. – (Національні стандарти України).
4. Соловйов В. І. Основи теорії надійності і експлуатації авіаційних систем / Соловйов В. І. – К. : КІ ВПС, 2000. – 248 с. – (Курс лекцій).
5. Ханджян А. О. Повышение надежности программного обеспечения информационно-измерительных и управляющих систем безопасности ядерных радиационно-опасных объектов : автореф. дис. на здобуття наук. ступеня канд. техн. наук : спец. 05.11.16 “Информационно-измерительные и управляющие системы” / А. О. Ханджян. – М., 2006. – 27 с.
6. Надёжность в технике. Основные понятия. Термины и определения : ГОСТ 27.002–89. – [Чинний від 01.07.1990]. – М. : Изд-во стандартов, 1990. – 24 с.
7. Курлаев С. А. Курс: Компьютерное моделирование : (электронні курси НГТУ) [Электронный ресурс] / Курлаев С. А., Волкова В. М., Гульяева Т.А. – Новосибирск : Новосибирский гос. техн. ун-т, 2013. – (Инст. дистанційного навчання) – Режим доступа : <http://dispace.edu.nstu.ru/didesk/course/show/3293>.
8. Підходи щодо оцінки надійності програмної компоненти головної системи єдиного інформаційного середовища / В. Л. Шевченко, Ю. А. Кіпрічников, В. А. Федорієнко [та ін.] // К: Збірник наукових праць ЦВСД НУОУ. – 2013. – №1(44) – С. 12–23.
9. Ермаков А. А. Основы надежности информационных систем / Ермаков А. А. – Иркутск. : ИрГУПС, 2006. – 151 с. – (Учебное пособие).
10. Перегуда А. И. Оценка показателей надежности автоматизированного технологического комплекса «Объект защиты – система безопасности» с нечеткими параметрами методом Монте-Карло [Электронный ресурс] / А. И. Перегуда, Д. А. Тимашов. // Информационные технологии – 2008. – №10. – С.7–15. – Режим доступа до журн. <http://library.mephi.ru/data/scientific-sessions/2010/fulltext/t5/4-7-1.doc>.