УДК 004.42:519.873

Р. В. Крепич[1], С. Я. Крепич[1]

КОМПЛЕКСНЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ЗБОРУ І ВІЗУАЛІЗАЦІЇ СТАТИСТИКИ ДЕФЕКТІВ ПРОГРАМНИХ ПРОЕКТІВ

[1]Тернопільський національний економічний університет

Анотація. В статті розглянуті вимоги до даних для побудови моделі надійності програмної системи. За допомогою UML діаграм класів описані процеси збору, трансформації (переведення з одного формату даних в інший), аналізу і візуального представлення інформації, щодо кількості зареєстрованих на проектах дефектів. Розроблений програмний комплекс, на основі поставлених вимог, надасть можливість проводити подальші дослідження моделювання надійності програмних систем з урахуванням кількості дефектів і часу їх реєстрації у відповідності до тривалості проекту. Планується провести детальний аналіз зібраних даних задля визначення закономірностей залежності кількості дефектів від різних характеристик проекту (складність архітектури, тривалість проекту, кількість тестувальників, кількість розробників). Більшість моделей надійності програмних систем використовують обмежену (не повну) кількість характеристик проекту, які впливають на надійність програмного продукту, тому планується проаналізувати існуючи моделі надійності програмних продуктів і перевірити їх на зібраних даних задля того, щоб визначити додаткові характеристики проекту, що потрібно враховувати для моделювання надійності програмного продукту.

Ключові слова: дефект, модель надійності програмного забезпечення, проект, база даних, густина дефекту.

Аннотация. В статье рассмотрены требования к данным для построения модели надежности программной системы. С помощью UML диаграмм классов описаны процессы сбора, анализа и визуального представления информации, по количеству зарегистрированных на проектах дефектов. Разработанный программный комплекс на основе поставленных требований, позволит проводить дальнейшие исследования моделирования надежности программных систем с учетом количества дефектов и времени их регистрации в соответствии с продолжительностью проект. Планируется провести детальный анализ собранных данных для определения закономерностей зависимости количества дефектов от различных характеристик проекта (сложность архитектуры, продолжительность проекта, количество тестировщиков, количество разработчиков). Большинство моделей надежности программных систем используют ограниченную (не полную) количество характеристик проекта, которые влияют на надежность программного продукта, поэтому планируется проанализировать существующие модели надежности программных продуктов и проверить их на собранных данных для того, чтобы определить дополнительные характеристики проекта, потриб- но учитывать для моделирования надежности программного продукта

Ключевые слова: дефект, модель надежности программного обеспечения, проект, база данных, плотность дефекта

Abstract. In the article the data requirements for construction of the reliability model of the software system are considered. Using UML class diagrams, we describe the processes of collecting, analyzing and visual representation of information on the number of defects registered on projects. The developed software package, based on the requirements, will provide the opportunity to conduct further research on the simulation of the reliability of software systems, taking into account the number of defects and the time of their registration in accordance with the duration of the project. It is planned to carry out a detailed analysis of the collected data in order to determine the patterns of dependence of the number of defects on the various project characteristics (complexity of the architecture, duration of the project, number of testers, number of developers). Most software system reliability models use a limited (not complete) number of project characteristics that affect the reliability of the software, so it is planned to analyze the existing software product reliability models and verify them on the collected data in order to determine the additional characteristics of the project, which is needed, but take into account for modeling the reliability of a software product

Key words: defect, software reliability model, project, database, defect densitydefect.
DOI: https://doi.org/10.31649/1999-9941-2018-42-2-35-42.

Вступ

На сьогоднішній день актуальною є проблема дослідження надійності програмної системи (ПС)[1]. Під надійністю програмної системи розуміємо здатність системи виконувати покладені на неї функції, протягом певного часу експлуатації в наперед визначених допустимих межах та за певних умов експлуатації[2-4]. Для багатьох систем надійність є основною характеристикою і критичною вимогою (системи реального часу, радарні системи, системи безпеки, медичне устаткування з вбудованими програмами та ін)[4-6], оскільки відмова такої системи або її неправильне функціонування призводить до серйозних наслідків, а час, необхідний для усунення дефектів системи у процесі її експлуатації, значно довший ніж під час перевірки програмної системи (етап тестування)[6].

Огляд особливостей математичних моделей надійності програмних систем[7-10] показав їх обмежене застосування, оскільки по своїй суті вони є статичними і відображають зв'язки між загальною кількістю кумулятивних дефектів у програмній системі. Взамін зазначеного класу моделей пропонується використовувати моделі динаміки, кумулятивних похибок у процесі проведення тестування, зокрема на стадії регресійного та інтегративного тестування. Однак перед тим, як перейти до моделювання надійності програмних систем за обраною схемою потрібно провести додатковий збір, аналіз і візуальне представлення реальних даних з загально доступних джерел, що є метою даної статті. Важливо провести збір даних з якомога більшої кількості різних джерел, щоб отримати якомога більшу можливість для аналізу, покращення існуючих моделей надійності ПС на основі кумулятивних помилок.

Мета дослідження

Розробка програмного забезпечення для збору даних з різних джерел, перевід отриманих даних в єдиний формат для подальшої візуалізації і аналізу на існуючих методах моделювання надійності програмних систем. Проведений аналіз дозволить вирахувати на основі великої кількості різноманітних даних додаткові коефіцієнти невраховані попередньо у загальновідомих моделях і тим самим підвищити їхню якість.

Постановка задачі

Основною метою збору і аналізу даних про дефекти проектів є отримання достатньої кількості інформації для розгляду і удосконалення моделей надійності програмного забезпечення, що відносяться до моделей кумулятивних помилок. Дані моделі використовуються для дослідження функціональної залежності інтенсивності відмов на помилку та інтенсивності відмов протягом певного періоду часу. Оскільки, здебільшого користувачеві не надається інформація про внутрішню структуру програмного продукту і висновки про його надійність можна зробити лише оцінивши його вхідні та вихідні дані, які можна використати для параметричної і структурної ідентифікації відомих моделей надійності програмних систем [7-10]. Тому у роботі доцільно розглядати моделі «чорної скриньки», предметом дослідження яких є кількість помилок у визначеному часовому інтервалі.

Найбільш поширеними моделями цього типу є деєутрофікаційна модель Jelinski та Moranda, модель Schick та Wolverton, геометрична деєутрофікаційна модель Jelinski та Moranda, геометрична пуассонова модель Moranda, модель недосконалого відлагодження Goel та Okumoto та ін[11].

Усі ці моделі базуються на наступних основних припущеннях [12]:

- Кумулятивна кількість відмов на момент t, $M(t)$ відповідає пуассоновому процесу з функцією математичного сподівання $\mu(t)$. При цьому, очікувана кількість помилок, які буде виявлено за проміжок часу $t + \Delta t$ пропорційна кількості помилок, що залишилися у системі на момент t. Також вважається, що $\mu(t)$ - обмежена, не спадаюча функція часу: $lim_{t\to\infty}\mu(t) = N < \infty$. Отже, ця модель належить до категорії скінчених функцій.
- Кількості помилок $f_1, f_2, \dots, f_n$, виявлених на відповідних часових інтервалах $[(t_0 = 0, t_1), (t_1, t_2), \dots, (t_{n-1}, t_n)]$ незалежні для будь якої скінченої кількості часових проміжків $t_1 < t_2 < \cdots < t_n$.
- Імовірності виявлення будь якої помилки є однаковими.
- При виявленні помилки, з програмного коду видаляється тільки один дефект без уведення нових.

При побудові моделі оцінювання та прогнозування надійності ПЗ кількість виявлених помилок розподілена за законом Пуассона. Динамічний показник складності проекту є параметром моделі та визначається на основі експериментальних даних і набуває значення додатного числа.

Функція інтенсивності виявлення помилок ПЗ прийме вигляд[12]:

$$\lambda(t) = \alpha\beta^{n+1}t^n\exp(-\beta t) \quad (1)$$

де α - коефіцієнт, який характеризує загальну кількість помилок, які були виявлені в програмі від початку спостереження, β - коефіцієнт, що характеризує швидкість зміни функції інтенсивності виявлення помилок, (завжди $\beta > 0$), n – параметр для оцінки величини проекту, де вибір параметру n залежить від процесу проведення тестування.

Для виразу (1) функція кумулятивної кількості помилок ПЗ має вигляд[12]:

$$\mu(t) = \int_0^t \lambda(\tau)d\tau = \alpha\left(n! - \sum_{i=0}^{n}\frac{n!\beta^{n-i}}{(n-1)!}t^{n-i}\,e^{-\beta t}\right) \quad (2)$$

Вирази (1) та (2) є моделлю з динамічним показником складності проекту.

Дана модель була вибрана для подальшого дослідження і покращення. З огляду на те, що на надійність ПС впливає багато чинників, такі як: термін, вартість, розмір, складність проекту, методологія його розробки і інші чинники, обґрунтованою є потреба перевірити існуючу модель надійності ПС на великій кількості даних різних проектів. Дана перевірка дасть можливість виявити, для яких проектів вибрана модель не є повноцінною і чому, тобто вона дасть можливість подальших досліджень додаткових параметрів, що варто враховувати при прогнозуванні надійності ПС, зав'язків між ними і їхню вагу в моделі.

Однак, для збору великої кількості даних потрібно розробити програмний комплекс, що спрямований на збір даних з різних ресурсів з різним форматом і їх представлення в єдиний формат для подальшого аналізу. Для збору даних було обрано загальнодоступні ресурси [13-16]

Тобто, поставленою задачею є розробка ПЗ, що дасть можливість збирати дані різного формату з багатьох джерел, подальше їх перетворення в єдиний формат для застосування в існуючій моделі, що дасть можливість проаналізувати існуючі параметри і коефіцієнти даної моделі, від чого вони залежать, а

також виявити додаткові параметри і коефіцієнти, введення яких в модель дасть можливість її покращення для більш точного прогнозування надійності ПС.

Вимоги до даних для побудови моделі

Одною з основних складових для побудови моделей є наявність достатньої кількості емпіричних даних, які в свою чергу, мають бути коректними і адекватними, в іншому разі буде не можливо побудувати потрібну модель, яка б змогла найточніше відтворювати реальні процеси і взаємодію даних. Тобто, одним з перших етапів побудови моделей є збір вхідних і вихідних даних, причому, кількість даних має бути якомога більшою, щоб включити якомога ширший спектр можливих сценаріїв внутрішніх процесів моделі.

В даному випадку, нас цікавить наступна інформація:

- проект і його основні атрибути (опис, основна ціль, початок і, можливо, завершення);
- інформація про дефекти проектів, що в свою чергу складається з
  - важливості дефекту, що означає на скільки дефект впливає на систему в загальному (для прикладу, якщо дефект спричиняє падіння системи, то його важливість критична). Дана інформація нам потрібна, щоб мати можливість працювати з дефектами в розрізі їхньої важливості і впливу на програмний продукт. Потрібно зазначити, що не завжди дефекти з високою важливістю будуть опрацьовані командою розробників в першу чергу. Тут більш важливий пріоритет дефекту, зокрема, якщо дефект має критичну важливість, але відтворюється вкрай складно або рідко, то він може отримати пріоритет меншого рівня і бути опрацьованим пізніше;
  - пріоритету дефекту, що означає на скільки дефект важливо виправити в плані часу. Тобто якщо дефект має високий пріоритет, то його потрібно опрацювати в першу чергу, бо він спричиняє некоректне функціонування важливих частин продукту;
  - дати реєстрації (тобто, коли дефект був знайдений і зареєстрований), щоб мати можливість проаналізувати частоту появи дефектів в часі (кількість знайдених дефектів на одиницю часу);
  - дата виправлення, щоб мати інформацію про час, який був потрібним для виправлення даного дефекту.
- більш детальна інформація про дефекти, що включає:
  - ким даний дефект був виявлений;
  - до якого модуля належить даний дефект;
  - детальний опис з кроками для відтворення;
  - людина, що має виправити даний дефект.

Оскільки для побудови точнішої моделі потрібні більш глибші (більша кількість однорідної інформації) та більш ширші (більша кількість атрибутів одної одиниці інформації) дані, доцільно розробити програмний продукту, що надасть можливість:

- працювати з декількома зовнішніми системами, що містять необхідну інформацію;
- отримувати з зовнішніх систем і тимчасово зберігати дані потрібної глибини і ширини в довільному форматі (формат даних відрізняється в залежності від системи, з якої їх отримано), який дасть змогу швидше і якісніше проаналізувати структуру даних;
- зберігати дані в кінцевому форматі в базу даних для подальшої зручної обробки;
- представляти збережені дані в різних форматах, для візуального аналізу, зокрема:
  - у вигляді таблиць з основною і додатковою інформацією;

у вигляді графіків різного спрямування (по часовому періоду і по типу графіку).

Вимоги до програмного забезпечення

З огляду на специфіку використання розроблюваного програмного забезпечення, поставлено наступні вимоги:

- підтримка декількох зовнішніх систем збору даних про дефекти проектів;
- можливість швидко збору, опрацювання і збереження даних;
- «портативність» - можливість використання програмного забезпечення на різних комп'ютерних станціях на основі операційної системи Windows без потреби встановлення додаткового програмного забезпечення. Щоб задовільнити дану вимогу пропонується використовувати базу даних SQLite, що не потребує встановлення серверної компоненти;
- зручний і зрозумілий інтерфейс з підтримкою української і англійської мови;
- наявність графічного відображення даних в декількох режимах;
- можливість налаштування поведінки програмного комплексу – зміни початкових налаштувань системи;
- сторінковий перегляд даних в табличному режимі – оскільки передбачається велика кількість даних, необхідною є умова можливості перегляду даних порціями;

- зберігання останніх вибраних змін користувача в базу даних (для зменшення кількості необхідних кроків користувача для отримання цікавої для нього інформації), зокрема:
    - збереження останнього вибраного користувачем проекту;
    - збереження вибраного часового режиму графіка (денний, тижневий, місячний);
    - збереження вибраного типу графіка (лінійний чи кумулятивний).

Спираючись на вище описане, визначимо, які основні функції програмного продукту будуть доступні користувачеві за допомогою діаграми варіантів використання представленої на рисунку 1.

Як видно з рисунку, користувач має можливість отримати дані про проекти і їх дефекти з різних джерел з можливістю збереження в базу даних та подальшим аналізом, який включає в себе візуалізацію даних в декількох представленнях

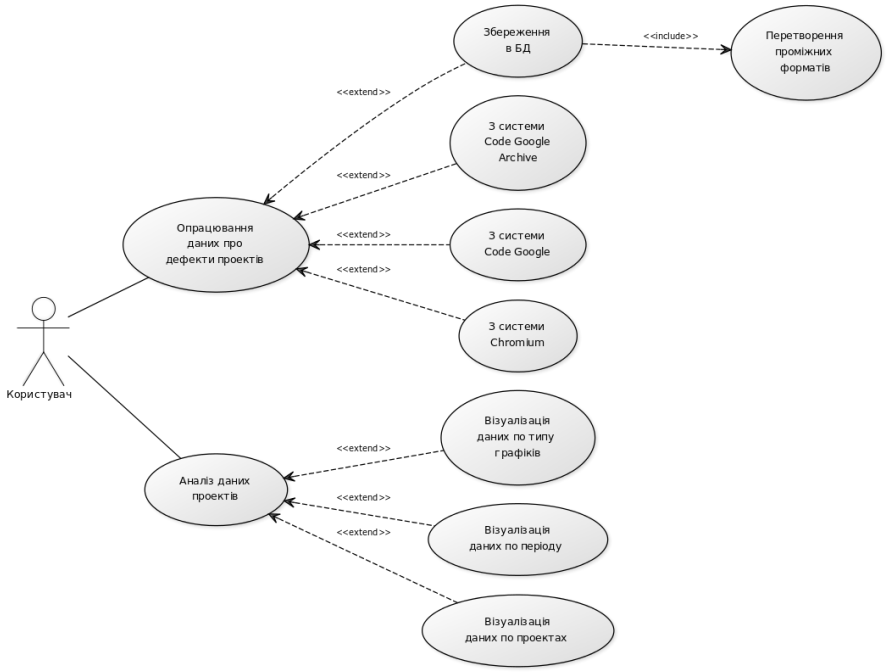

Рисунок 1 - Діаграма варіантів використання розроблюваного програмного комплексу

Архітектура програмного забезпечення

Коротко розглянемо структуру модуля отримання даних. На рисунку 2 наведена діаграма компонентів модуля отримання даних з різних систем і переведення їх в єдиний формат. Оскільки дані в кожній системі представлені в різному вигляді, потрібно розробити компонент, що буде розпізнавати потрібні дані для кожної системи. Даний компонент розпізнавши потрібні дані зберігає їх в проміжному форматі, зручному для аналізу і приведення до єдиного формату [17].

Рисунок 2 - Діаграма компонентів модуля отримання даних і переводу в єдиний формат

На рисунку 3 наведена діаграма основних класів, які використовуються для отримання даних з зовнішніх систем. Оскільки кожна система є унікальною, тому для кожної системи розроблений окремий клас, що містить всі необхідні методи для отримання даних. Дані класи наслідуються від базового класу, який містить загальні методи і властивості для всіх систем

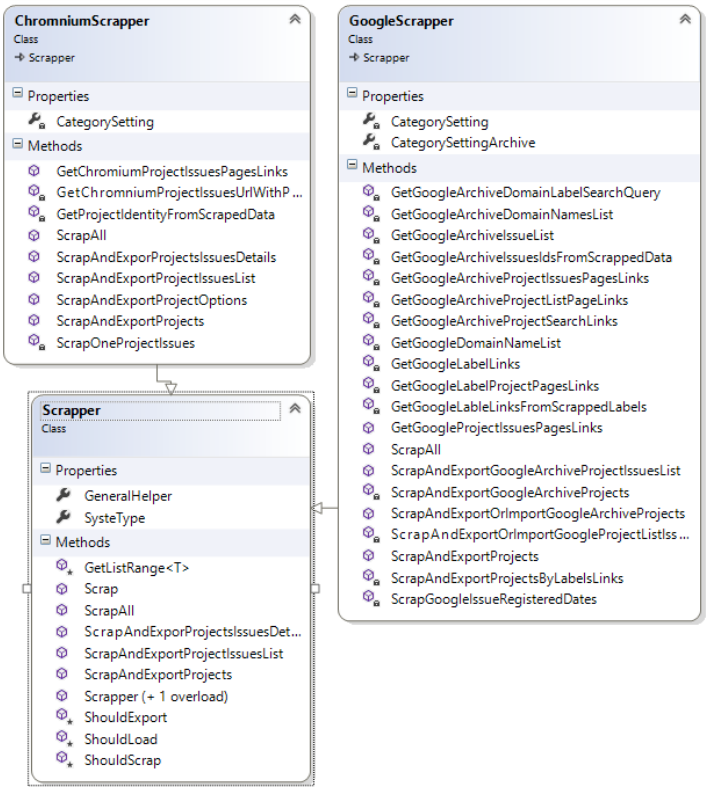

Рисунок 3 - UML діаграма основних класів модуля отримання даних

Приклад застосування

Розглянемо один із можливих сценаріїв роботи користувача з розробленим програмним продуктом збору, аналізу і візуалізації даних.

На першому етапі користувачеві потрібно отримати дані проекту, який його цікавить. Для цього він може скористатися меню програмного комплексу, що дає йому можливість отримати і імпортувати дані про проект і його дефекти в базу даних (див.рис.4)

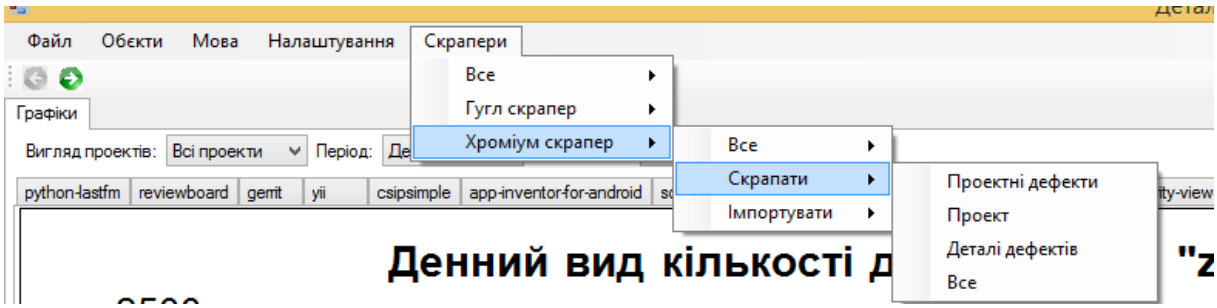

Рисунок 4 - Інтерфейс представлення можливості отримання/імпортування даних

Отримавши дані з зовнішної системи, користувач має можливість переглянути їх в табличному вигляді, який є зручним, коли потрібно переглянути деталі отриманих даних. На рисунку 5 представлене табличне відображення даних.

Також користувач має можливість переглянути дані про дефекти проекту у вигляді лінійного графіку. На рисунку 6 представлене графічне відображення інформації про кількість дефектів виявлених в проекті до поточного моменту часу, що дає користувачеві можливість відслідкувати пікові значення кількості дефектів.

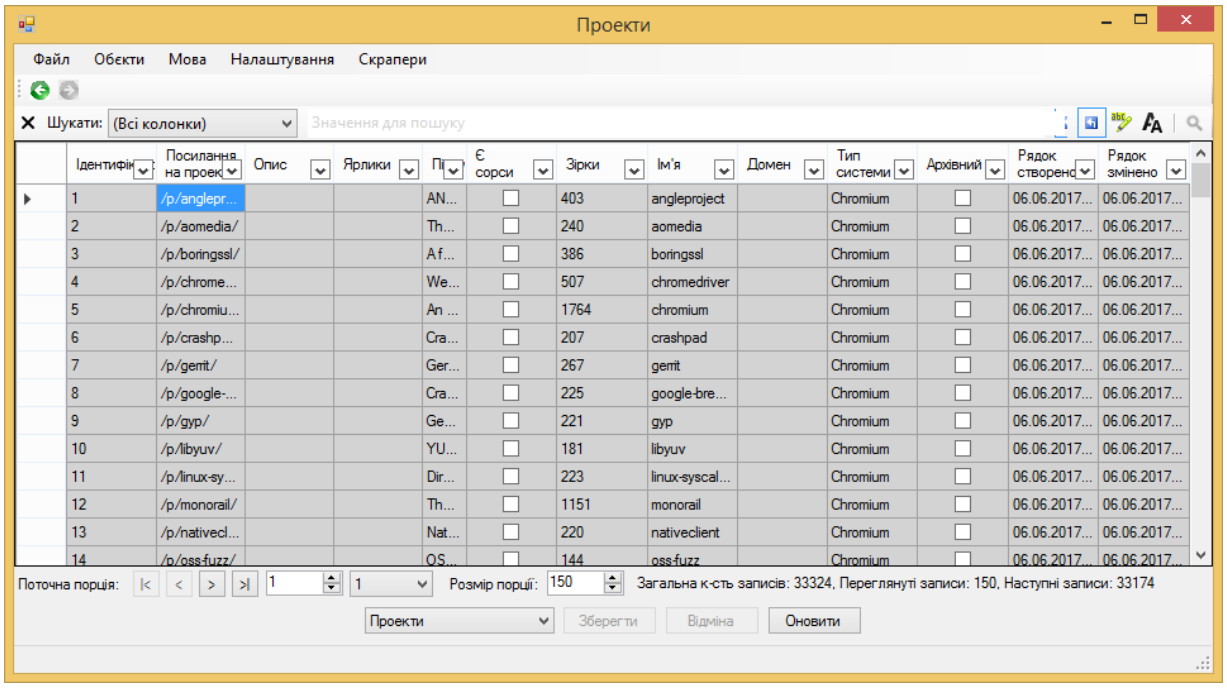

Рисунок 5 – Табличне представлення отриманих даних

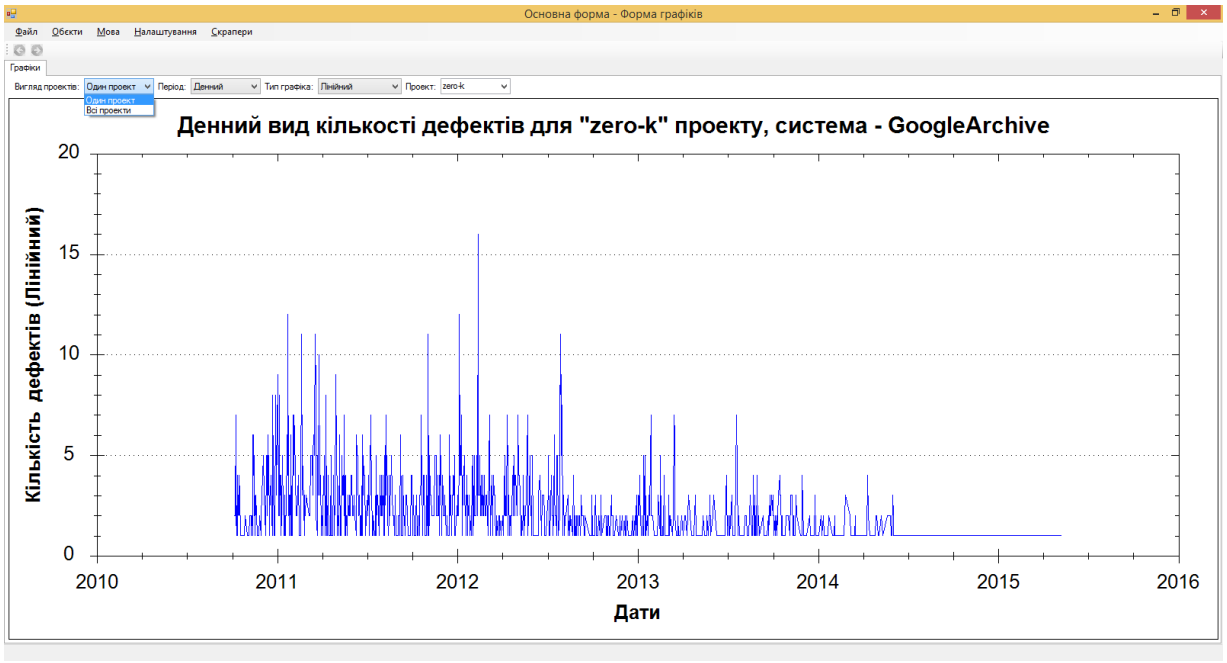

Рисунок 6 – Графічне відображення інформації про дефекти проекту в лінійному представленні

Користувач має можливість переключити графічне відображення даних з лінійного на акомулятивне. Дане представлення показано на рисунку 7. Тут користувач може побачити кількість дефектів за певний період часу, який також може бути змінений (денний, тижневий або місячний), що дає йому можливість відслідкувати стабілізацію кількості дефектів на проекті.

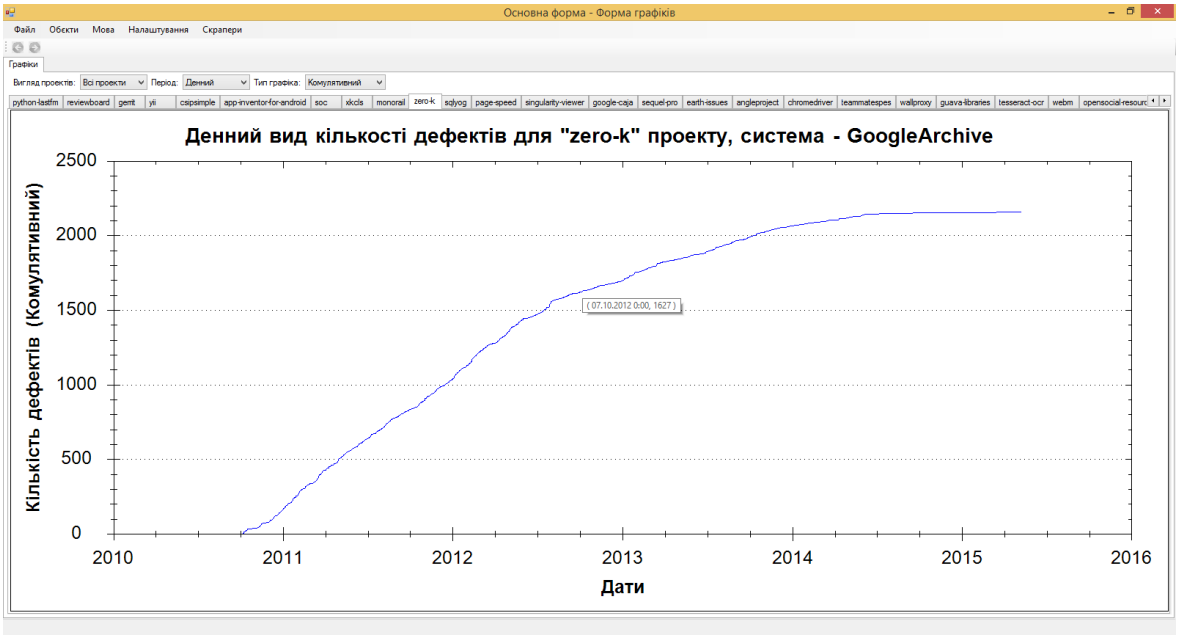

Рисунок 7 – Графічне відображення інформації про дефекти проекту в акомулятивному вигляді

Висновки

На сьогоднішній день моделі оцінювання надійності програмних систем здебільшого базуються (розроблені) на однотипних даних зібраних з одного джерела. У даній роботі описано потреба, вимоги і архітектура програмного комплексу, основною ціллю якого являється збір даних з різноманітних джерел, перетворення до єдиного формату і візуалізація даних про дефекти проектів для подальшого їх аналізу, перевірки і адаптації на відомих математичних моделях, визначення надійності системи на основі кумулятивної похибки.

Оскільки даний програмний комплекс спрямований на збір даних з різних джерел, це дасть можливість отримати потрібну, більш повноцінну інформацію для перевірки і адаптації існуючих методів моделювання надійності програмних систем на основі кумулятивної похибки.

Зібрана інформація буде використана в подальших дослідженнях для удосконалення відомих моделей надійності програмних систем за рахунок визначення додаткових коефіцієнтів, які дозволять використовувати вказані моделі на будь яких наборах вхідних даних.

Список літератури

[1] *ДСТУ 2844-94. Програмні засоби ЕОМ. Забезпечення якості. Терміни та визначення*. К. : Держстандарт України, 1996. – 19 с.

[2] H. Pham, System software reliability, Springer-Verlag: London Limited, 2006. – 440 p.

[3] S. Krepych, A. Dyvak, M. Dyvak, I. Spivak, "The method of providing of functional suitability of elements of the device of formation of signal in electrophysiological way of classification tissues surgical wound," on *13th International Conference Perspective Technologies and Methods in MEMS Design*, MEMSTECH 2017 Proceedings, pp. 183-186.

[4] С. Я. Крепич, «Метод синтезу смугового фільтра для заданих обмежень на його модуль коефіцієнта передачі,» на *Сучасні комп'ютерні інформаційні технології: Матеріали IV Всеукраїнської школи-семінару молодих вчених і студентів АСІТ'2014*. – Тернопіль: Економічна думка, 2014. – с. 26-29.

[5] S. Krepych, P. Stakhiv, and I. Spivak, «Analysis of the tolerance area parameters REC based on technological area scattering» on *12-th International Conference "The Experience Of Designing And Application Of CAD Systems in Microelectronics*". – 2013. – pp. 179–180.

[6] Р. В. Крепич «Особливості підходів до моделювання надійності програмних систем,» на VI Всеукраїнській школі-семінарі молодих вчених і студентів АСІТ'2016, 2016. с. 125–126.

[7] R. Peng, Y. F. Li, W. J. Zhang, and Q. P. Hu, Testing effort dependent software reliability model for imperfect debugging process considering both detection and correction *Reliability Engineering and System Safety*. Vol. 126, p. 37–43, 2014.

[8] С. Я. Крепич «Моделювання та забезпечення функціональної придатності статичних систем методами аналізу інтервальних даних,» дис. канд. техн. наук, НУ «Львівська політехніка», Львів, 2016. 166 с.

[9] M.Dyvak, P. Stakhiv, I. Maksymova, and O. Potravych «Identification of the dynamic models by the adaptive method of tolerance estimation,» on *The Experience of Designing and Application of CAD Systems in Microelectronics - Proceedings of the 9th International Conference, CADSM 2007*, 2007. p. 365–368.

[10] М. П. Дивак, *Задачі математичного моделювання статистичних систем з інтервальними даними*, Тернопіль: Економічна думка, 2011. – 216 с.

[11] Б. В. Гнеденко, Ю. К. Беляев, и А. Д. Соловьев, *Математические методы в теории надежности*, М.: "Наука", 1965. – 524 с.

[12] В. С. Яковина «Методи та засоби аналізу надійності функціонування програмного забезпечення з урахуванням етапів життєвого циклу» : дис. докт. техн. наук, Львів, 2016. – 325 с.

[13] Monorail [Online] Available: https://bugs.chromium.org/

[14] Google Code [Online] Available: https://code.google.com/

[15] [Online] Available: https://storage.googleapis.com/google-code-archive/v2/

[16]System Dashboard [Online] Available: https://bugreports.qt.io/secure/Dashboard.jspa

[17] Э. Троелсен, *Язык программирования C# 6.0 и платформа .NET*, Диалектика-Вильямс. 2016. 1440 с.

Стаття надійшла: 28.05.2018.

Відомості про авторів

Крепич Роман Володимирович – аспірант кафедри комп'ютерних наук,
Крепич Світлана Ярославівна – к. т .н., старший викладач кафедри комп'ютерних наук.

R. V. Krepych[1], S. Y. Krepych[1]

COMPLEX SOFTWARE FOR COLLECTION AND VISUALIZATION OF SOFTWARE DEFECTS STATISTICS

[1]Ternopil National Economic University