DOI 10.31891/2307-5732-2021-295-2-64-70
УДК 004.052; 004.416.2

В.С. ЯКОВИНА, Б.В. УГРИНОВСЬКИЙ
Національний університет «Львівська політехніка»

ДОСЛІДЖЕННЯ СИСТЕМНИХ ПРОЦЕСІВ ТА КОРИСТУВАЦЬКИХ ДОДАТКІВ ОПЕРАЦІЙНОЇ СИСТЕМИ ANDROID В КОНТЕКСТІ ЯВИЩА СТАРІННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В роботі наведено результати досліджень явища старіння програмного забезпечення в операційній системі Android. Виконано аналіз оцінки oom_adj_score та на її основі виділено дві групи: системні процеси та користувацькі додатки. Аналіз системних процесів показав, що індикаторами старіння є процеси system_server та surfaceflinger, а також com.android.phone, cameraserver у випадку активного використання додатків телефонної книги та камери. В свою чергу, користувацькі додатки також вразливі до явища старіння, що спостерігається в можливих витоках пам'яті та помилках відображення кадрів UI. Запропоновано застосувати розглянуті системні процеси в процедурі омолодження програмного забезпечення, а для користувацьких додатків реалізувати засоби, які б забезпечили розробників інформацією про стан старіння системи для можливості побудови алгоритмів з урахуванням цього стану.

Ключові слова: старіння програмного забезпечення, метрики старіння, омолодження програмного забезпечення, операційна система, Android.

VITALIY S. YAKOVYNA, BOHDAN V. UHRYNOVSKYI
Lviv Polytechnic National University

RESEARCH OF SOFTWARE AGING IN ANDROID SYSTEM PROCESSES AND USER APPLICATIONS

Android operating system is vulnerable to the aging-related effects such as performance degradation and increased of aging-related failures rate due prolonged usage of a mobile device without rebooting. This paper considers software aging phenomenon in system processes and user applications of the Android operating system and means for counteracting this phenomenon. Experimental research was performed using a methodology that consists in performing stress tests on mobile applications, collecting system data on running processes, converting the collected data into time series for the relevant metrics and analyzing these data using statistical methods. Thus, the analysis of oom_adj_score for determining processes priorities in the context of software aging allowed to identify two groups of processes, namely system processes and user applications. It is also pointed out the possibility of using oom_adj_score to determine the state of system usage in the tasks of software aging predicting and performing software rejuvenation. The results of the system processes analysis showed that the indicators of aging are system_server and surfaceflinger processes, as well as com.android.phone, cameraserver in the case of active use of contacts and camera applications. The considered processes can be used to implement software rejuvenation. Research has shown that user applications are also vulnerable to aging-related effects, but the rejuvenation procedure cannot be applied to them at the system level. It is important to take steps to prevent aging-related errors, such as using appropriate data structures and algorithms for efficient memory management, minimizing the load on the main UI stream, and using effective graphics techniques to reduce the number of delayed frames. In future works it is important to investigate the considered system processes and services in tasks of software aging forecasting and performing of rejuvenation procedure. It is important for user applications to develop tools that provide developers with information about the state of software aging in the system, which would allow to decide on the feasibility of performing important and resource-intensive tasks in conditions when the system is already in a state with a high probability of aging-related failure.

Keywords: software aging, aging metrics, software rejuvenation, operating system, Android.

Постановка проблеми

Старіння програмного забезпечення [1–4] – це процес погіршення продуктивності та збільшення частоти відмов програного забезпечення в системах, що працюють тривалий час без перезавантаження. Причиною виникнення цього явища є помилки, які накопичуються в системі та користувацьких додатках під час їх виконання, але відмова програмного забезпечення чи вплив цих помилок на продуктивність може спостерігатися пізніше їх виникнення. Прикладом таких помилок є витоки пам'яті чи помилки заокруглення. Явище старіння проявляється в багатьох системах [3, 4], зокрема в мобільних операційних системах, таких як Android [5, 6]. Мобільні пристрої є особливо вразливі до ефектів старіння [7], оскільки вони мають обмежені апаратні ресурси, залежні від заряду батареї та використовуються тривалий час без вимкнення.

Можна виділити два основні підходи для протидії явищу старіння, а саме активний та пасивний підхід.

Омолодження програмного забезпечення [2, 8] – це активний підхід для боротьби із явищем старіння, який полягає у виконанні перезавантаження чи очищення процесів, системних компонент чи всього девайсу в запланований час, щоб зменшити кількість накопичених помилок старіння в стані системи і тим самим відтермінувати виникнення відмов через старіння. Системні процеси, сервіси та користувацькі додатки можуть розглядатися як об'єкти спостереження для виявлення старіння та планування процедури омолодження, так і як цільові об'єкти для виконання перезавантаження чи очищення.

У свою чергу, пасивний підхід полягає у використанні інструментів на етапі розробки програмного забезпечення та застосуванні технік та засобів [9, 10] в самому програмному забезпеченні під час його виконання, що можуть зменшити інтенсивність виникнення помилок старіння.

Таким чином, важливими науково-прикладними задачами є визначення процесів, що можуть бути застосовані під час омолодження програмного забезпечення, виявлення умов виникнення старіння для

різних типів процесів, а також визначення можливих засобів для уникнення помилок старіння.

Аналіз останніх джерел

Дослідження явища старіння в першу чергу полягає у виявленні чинників та умов старіння, проблемних ділянок операційної системи та користувацьких додатків, які піддаються впливам старіння. Для отримання експериментальних даних та їх подальшого аналізу в більшості робіт застосовується методологія [7], яка полягає у виконанні стресових тестів над додатками операційної системи Android, регулярному збиранні даних про стан системи під час виконання стресового тестування, формуванні часових рядів для певного набору метрик на основі зібраних даних, аналізі часових рядів статистичними та регресійними методами. Результати аналізу дозволяють визначити ефективні та достовірні метрики старіння, умови та ділянки системи, де виникає старіння, а також виявити потенційних кандидатів для виконання процедури омолодження.

В роботах [5, 6] виконано грунтовне дослідження явища старіння в операційній системі Android, виявлено основні характеристики старіння на рівні операційної системи та запропоновано засоби омолодження шляхом перезавантаження основних системних сервісів. Автори враховували різні умови та чинники, що могли впливати на старіння, зокрема, модель та технічні характеристики мобільного пристрою, версію операційної системи, різні набори користувацьких додатків, події введення, запуски і зупинки користувацьких додатків, обсяг вільної пам'яті в сховищі даних, тощо. В дослідженнях розглядали системні метрики та метрику користувацького інтерфейсу, зокрема, наступні:

- PSS (Proportional Set Size, kB) – частка оперативної пам'яті, що зайнята певним процесом і складається з приватної пам'яті цього процесу та частки спільної пам'яті одного чи декількох процесів.
- GC Total Time та GC Paused Time (ms) – загальна тривалість виконання збирача сміття (garbage collector) і затримка перед її виконанням.
- Activity Launch Time (ms) – це кількість часу, що пройшло від моменту запуску процесу до остаточного відображення відповідного Activity (екрану додатку) на дисплеї мобільного пристрою.

Результати досліджень [5] показують, що найбільш вразливими і ресурсоємними процесами системи є:

- System Server – це Java процес, який запускається під час завантаження Android та ініціалізує Android Framework. Він містить більшість системних служб, таких як Activity Manager, яка управляє життєвим циклом додатків та їх Activity, та Package Manager, яка управляє установленими пакетами та дозволами безпеки. Цей процес також регулює доступ до системних ресурсів.
- System UI – це процес, який компонує області екрана для відображення сповіщень, повідомлення про стан пристрою та кнопок навігації за допомогою системних панелей.
- Surface Flinger – це процес, що отримує шари вікон (поверхні) з різних джерел (включно із System UI), об'єднює їх та відображає на дисплеї девайсу.

В попередніх роботах основна увага приділяється дослідженню старіння на рівні операційної системи, тому важливо розглянути аспекти старіння процесів різного рівня, зокрема користувацьких додатків. В роботі [11] розглянуто метрики Frame Draw Time та Janky Frames Count, які дозволяють відслідковувати явище старіння в користувацьких додатках. Frame Draw Time – це час відображення одного кадру інтерфейсу користувача. Janky Frames Count – це кількість кадрів, які не відобразились через затримку внаслідок навантаження. Команда командного рядка *adb shell dumpsys gfxinfo* дозволяє отримувати описані метрики, а також показники, що вказують на можливі причини затримок кадрів чи тривалого відображення кадрів [12]:

- Missed Vsync – пропущені вертикальні синхронізації відображення кадру і оновлення дисплею девайсу;
- High input latency – висока затримка введення для відображення наступного кадру;
- Slow UI thread – повільна робота потоків користувацького інтерфейсу;
- Slow bitmap uploads – повільне завантаження растрових зображень;
- Slow issue draw commands – повільне виконання команд малювання.

Операційна система Android реалізує два механізми для управління випадками низького рівня об'єму вільної оперативної пам'яті [13]: kernel swap daemon (KSD) та low-memory killer (LMK). На рівні процесів системи використовується LMK, який «вбиває» процеси, тобто примусово завершує їх роботу і звільняє використовувані ресурси. Для прийняття рішення про те, який процес може бути завершений, LMK використовує оцінку "out of memory" (oom_adj_score) для пріоритезації кожного процесу системи. Процеси з найвищою оцінкою примусово завершуються першими. Основні категорії оцінок процесів:

- Фонові додатки (cached, serviceb) – не активні в даний момент додатки. Завершуються першими в порядку зменшення присвоїного їм рейтингу.
- Попередній додаток (prev) – нещодавно запущений додаток, який з більшою імовірністю користувач знову відкриватиме.
- Домашній додаток (home) – програма запуску, закриття якої призведе до зникнення шпалер на головному екрані.
- Сервіси (servicea) – це сервіси додатків, які запускаються і контролюються в першу чергу додатками і можуть виконувати, наприклад, завантаження даних на сервер чи інші складні обчислення.
- Помітні додатки (percept) – це помітні для користувача додатки, але не знаходяться на

передньому плані, наприклад, системні кнопки навігації.

- Додаток переднього плану (fore, vis) – це додаток, що відображається і використовується безпосередньо користувачем в даний момент. Примусове завершення цього додатку виглядатиме для користувача як помилка виконання і є індикатором того, що девайс працює погано.
- Стійкі сервіси (pers, persvc) – основні сервіси мобільного пристрою, які управляють телефонним, wifi зв'язком і т.д..
- Системні процеси (system) – у випадку завершення цих процесів телефон може перезавантажитись.
- Нативні процеси (native) – низькорівневі процеси системи, зокрема KSD.

Метою роботи є дослідження явища старіння в системних процесах та користувацьких додатках операційної системи Android, враховуючи метрики Frame Draw Time, Janky Frames Ratio та пов'язані з ними причини затримки відображення кадрів, а також пріоритети процесів oom_adj_score.

Виклад основного матеріалу

Розроблений в попередній роботі фреймворк [11] для виконання стресового тестування мобільних додатків операційної системи Android та формування часових рядів для метрик старіння, на основі зібраних системних даних, дозволив отримати експериментальні дані для аналізу старіння системних процесів та користувацьких додатків. Детальна інформація про виконувані тести та вимірювані метрики подана в таблиці 1.

Таблиця 1

Конфігурація експериментальних досліджень

Умови стресового тестування	К-сть тестів	Вимірювані метрики	Користувацькі додатки	Конфігурація девайсу	Тривалість тестового запуску
Примусовий перезапуск додатків користувача кожні 30 секунд	4	***UI:*** Activity Launch Time, Frame Draw Time, Janky Frames Count, Missed Vsync, High input latency, Slow UI thread, Slow bitmap uploads, Slow issue draw commands; ***RAM:*** PSS, oom_adj_score; ***GC:*** Paused Time, Total Time;	com.google.android.youtube com.android.contacts com.android.chrome com.android.camera com.google.android.apps.photos	Xiaomi Redmi Note 4 2018 RAM: 3 Gb Версія Android: 7.0	2,5 год
Без примусового перезапуску додатків користувача	4				

Результати виконання експериментів

Виконання експериментального етапу дослідження дозволило отримати дані про використання процесами пам'яті системи, а також про динаміку зміни продуктивності різних процесів та додатків. Розподіл використання пам'яті процесами різних категорій (рис. 1) показує, що більша частина оперативної пам'яті використовується користувацькими додатками та сервісами: pers, fore, vis, home, prev, cached. В цьому випадку важливо ідентифікувати групи процесів, в яких спостерігається стабільний тренд збільшення використання оперативної пам'яті, що може свідчити як про наявні в них помилки пов'язані із витоками пам'яті, так і про природнє збільшення використовуваної пам'яті для користувацьких потреб.

На рис. 2 показано кількість трендів збільшення використання пам'яті процесами різних груп. Визначення тренду виконано з допомогою тесту Манна-Кендала [14] та процедури Сена для обчислення нахилу тренду. Статистична значимість для виявлення тренду визначена як 90% (α=0.1). Тобто, якщо $P < 0.1$, а значення нахилу більше нуля, тоді спостерігається тренд збільшення. Результати обчислення трендів показують, що майже у всіх випадках тестів спостерігається стабільне збільшення використання пам'яті користувацькими додатками (vis) та збільшення об'єму кешованих додатків (cached, backup), що свідчить про збільшення навантаження на систему і необхідність в майбутньому використання LMK для звільнення пам'яті для нових процесів. Недоліком такої ситуації є те, що прийняття рішення про завершення роботи користувацьких процесів відбувається на стороні системи, що в деяких випадках може бути критично для користувача, який очікує завершення виконання своїх задач. Категорія pers, в яку входять процеси системних сервісів, також зазнає старіння в 7 з 8 тестів.

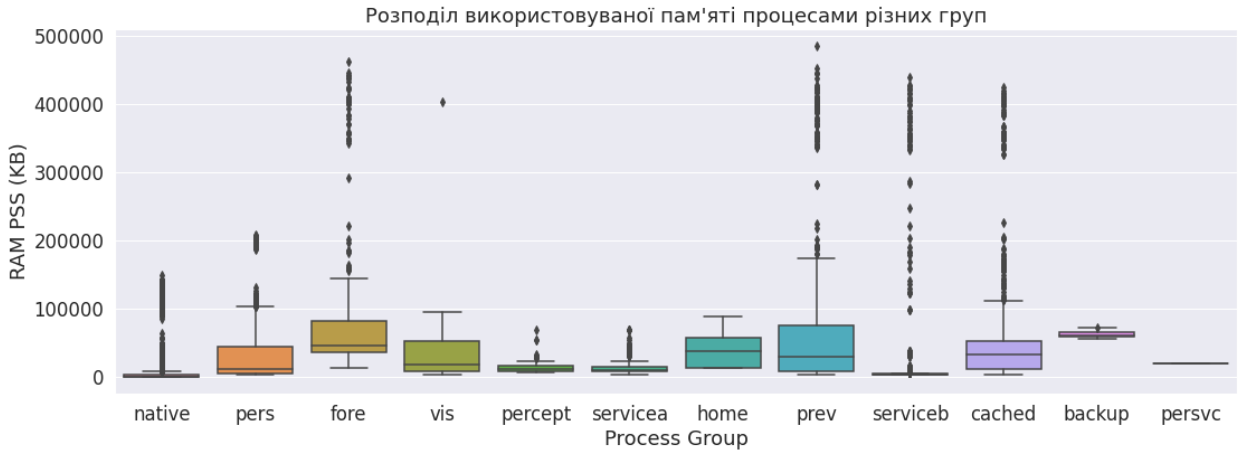

Рис. 1. Розподіл використання пам'яті процесами в різних групах

Рис. 2. Тренди використання оперативної пам'яті процесами різних груп

Проаналізувавши отримані результати і беручи до уваги роботу LMK, можна виділити дві умовні групи процесів з точки зору старіння програмного забезпечення:

1. Системні процеси (native, system, pers, persvc, home, percept) – можна застосувати процедуру омолодження до основних сервісів системи;

2. Користувацькі додатки (fore, vis, prev, servicea, cached, serviceb) – ефективність протидії старінню в цих процесах полягає в першу чергу в мінімізації кількості помилок, що призводять до старіння, шляхом застосування ефективних структур даних, алгоритмів, архітектурних шаблонів та механізмів управління ресурсами.

Аналіз oom_adj_score також дозволяє зробити висновок про доцільність використання цієї метрики у випадку визначення стану системи під час планування чи виконання процедури омолодження ПЗ. Визначення кількості користувацьких додатків, що відображаються на передньому плані та сервісів, що виконуються в фоновому режимі, дозволяє приймати зважене рішення про доцільність виконання процедури омолодження, щоб уникнути перешкоджання використання девайсу користувачем через перезавантаження тих чи інших процесів.

Аналіз системних процесів

Для аналізу старіння системних процесів виконано підрахунок кількості трендів збільшення використання пам'яті процесами та збільшення тривалості роботи збирача сміття серед всіх виконаних тестів з допомогою методу Манна-Кендала. На рисунку 3 зображено тільки ті процеси, в яких спостерігається старіння більше ніж в половині виконаних тестів і середній обсяг використовуваної оперативної пам'яті перевищує 20 Mb. Процес *system* показує найбільшу вразливість до старіння. В інших процесах також регулярно спостерігаються тренди збільшення використання пам'яті, однак відсутні тренди тривалості чи затримки роботи збирача сміття.

Серед розглянутих системних процесів, в яких спостерігаються тренди старіння, не виявлено сильних кореляцій із метриками відображення кадрів. Спостерігається незначна негативна кореляція між *PSS cameraserver* та *Janky Frames Ratio*, яка коливається між -0,12 і -0,19 для 7 із 8 стресових тестів. Така кореляція може пояснюватися тим, що збільшення об'єму пам'яті, яку використовує процес, дозволяє зменшити кількість затримок відображення кадрів. Припускається, що наявність чи відсутність кореляцій може залежати від конфігурації стресового тесту. Зокрема, в цій роботі стресовому навантаженню піддавався додаток камери, який використовувався приблизно 0,2 від всього часу виконання тестів. Внаслідок цього спостерігається постійний тренд збільшення використання пам'яті системним сервісом

cameraserver та кореляція із метрикою частки пропущених кадрів.

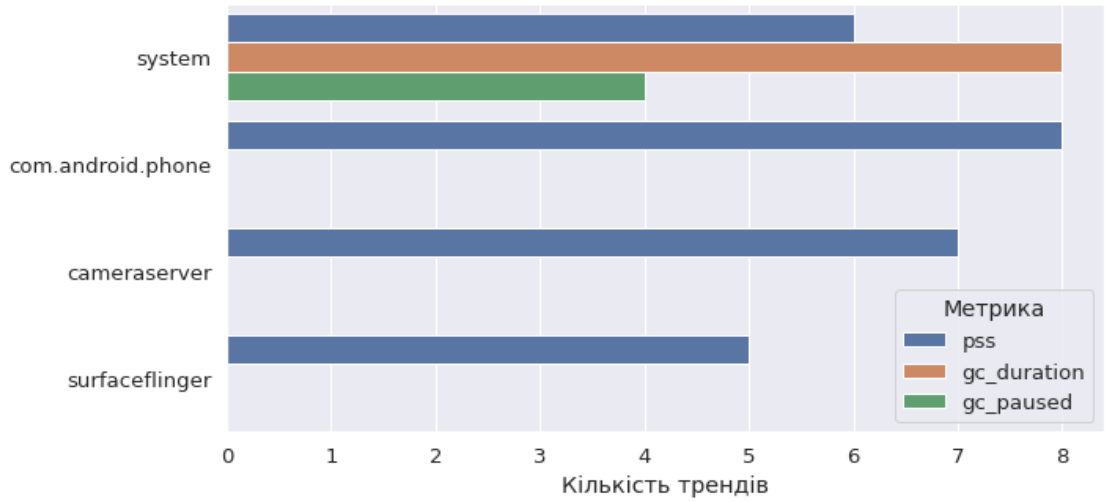

Рис. 3. Системні процеси Android, які найчастіше проявляють статистично значимі тренди збільшення використання пам'яті та збільшення тривалості роботи GC

Таким чином, аналіз системних процесів дозволяє зробити висновок, що в операційній системі Android ефектам старіння піддаються процеси *system* та *surfaceflinger*, а також процеси, які в більшій мірі залежать від діяльності користувача та роботи користувацьких додатків. Наприклад, *com.android.phone, cameraserver*, *audioserver* та інші. Тому, для реалізації засобів протидії старінню чи омолодження програмного забезпечення важливо дослідити і враховувати вплив різних процесів в залежності від конкретних умов використання мобільного пристрою.

Аналіз користувацьких додатків та фонових сервісів

Серед більшості користувацьких додатків та процесів спостерігаються тренди збільшення використання пам'яті, що є природньо. Трендів збільшення тривалості роботи збирача сміття не виявлено. В одному випадку з восьми було виявлено тренди збільшення метрик Frame Draw Time та Janky Frames Ratio для процесу com.android.chrome. Отримані результати можуть пояснюватись короткою тривалістю виконання стресових тестів.

Для уникнення старіння в користувацьких додатках важливо розуміти причини затримки кадрів та збільшення часу їх відображення (рис. 4). Виконані експерименти показують, що найбільше кадрів втрачається внаслідок перевантаження основних потоків відображення UI, пропусків вертикальної синхронізації, а також внаслідок повільного виконання команд малювання. Таким чином, можна зробити висновок, що зменшення впливу розглянутих чинників дозволить зменшити ефекти старіння загалом, тому важливо враховувати рекомендації, що стосуються уникнення виконання складних обчислень в основному потоці UI, ефективного використання методів малювання графіки, растрових зображень та інше.

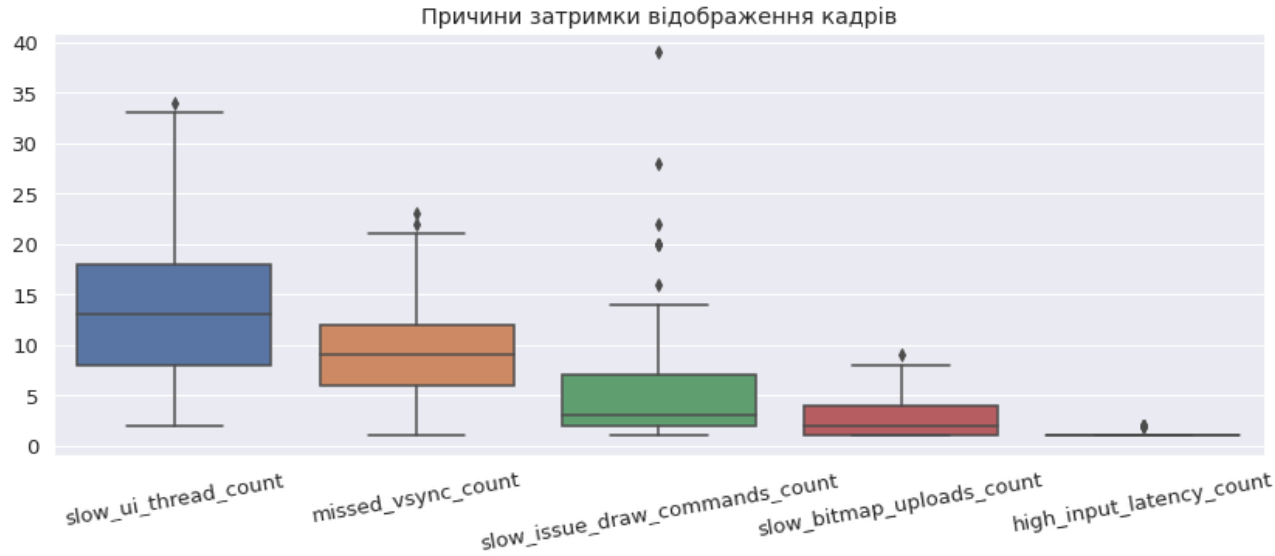

Рис. 4. Причини затримки відображення кадрів UI для виконаних експериментів

Висновки

Експериментальним шляхом досліджено явище старіння системних та користувацьких процесів

операційної системи Android.

Виконано аналіз оцінки процесів oom_adj_score з точки зору старіння програмного забезпечення. Розглянута оцінка може бути застосована для визначення стану використання системи користувачем в методах прогнозування старіння та виконання процедури омолодження. Визначено дві умовні групи процесів за oom_adj_score, а саме групи системних та користувацьких процесів.

Визначено, що системні процеси system_server та surfaceflinger можуть бути індикаторами старіння в системі, а такі процеси як cameraserver, audioserver можуть бути індикаторами в залежності від сценарію використання мобільного пристрою користувачем. Розглянуті процеси можуть бути застосовані для реалізації омолодження програмного забезпечення.

Користувацькі додатки також вразливі до явища старіння, однак процедура омолодження для них не може бути застосована так, як до процесів на рівні системи. Тому, важливо вживати заходів для попередження виникнення помилок старіння, а саме, використовувати відповідні структури даних та алгоритми для ефективного управління пам'яттю, мінімізувати навантаження на основний потік UI та використовувати ефективні методи малювання графіки, щоб зменшити кількість затриманих кадрів.

В майбутніх роботах планується дослідження системних процесів та сервісів в задачах прогнозування старіння та виконання процедури омолодження. Для користувацьких додатків важливо розробити засоби, які б забезпечували розробників інформацією про стан старіння системи, що дозволило б приймати рішення про доцільність виконання важливих і ресурсомістких задач в умовах, коли система вже в стані з високою ймовірністю виникнення відмови через старіння.

Література

1. Parnas D. L. Software aging / D. L. Parnas // Proceedings of 16th International Conference on Software Engineering. – 1994. – P. 279-287. URL: https://doi.org/10.1109/ICSE.1994.296790

2. Huang Y. Software rejuvenation: analysis, module and applications / Y. Huang, C. Kintala, N. Kolettis, N. Fulton // Proceedings of Twenty-Fifth International Symposium on Fault-Tolerant Computing. – 1995. – P. 381–390. URL: https://doi.org/10.1109/FTCS.1995.466961

3. Dohi T. Handbook of software aging and rejuvenation / T. Dohi, K. Trivedi, A. Avritzer // World Scientific Publishing Co Pte Ltd. – 2020. – P. 424. URL: https://doi.org/10.1142/11673

4. Grottke M. The fundamentals of software aging / M. Grottke, R. M. Jr, K. S. Trivedi // IEEE International Conference on Software Reliability Engineering Workshops. – 2008. – P. 1-6. URL: https://doi.org/10.1109/ISSREW.2008.5355512

5. Cotroneo D. Software aging analysis of the android mobile os / D. Cotroneo, F. Fucci, A. K. Iannillo, R. Natella, R. Pietrantuono // IEEE 27th International Symposium on Software Reliability Engineering. – 2016. – P. 478-489. URL: https://doi.org/10.1109/ISSRE.2016.25

6. Cotroneo D. A Comprehensive Study on Software Aging across Android Versions and Vendors / D. Cotroneo, A. K. Iannillo, R. Natella, R. Pietrantuono // Empirical Software Engineering. – 2020. – Volume 25(3). - P. 3357-3395.

7. Яковина В. С. Старіння програмного забезпечення мобільних додатків: аналіз проблематики / В.С. Яковина, Б. В. Угриновський // Науковий вісник НЛТУ України. – № 30(2). – С. 107–112. URL: https://doi.org/10.36930/40300219

8. Cotroneo D. A Configurable Software Aging Detection and Rejuvenation Agent for Android / Cotroneo D., Simone, L. D., Natella, R., Pietrantuono, R., Russo, S. // 11th Intl Workshop on Software Aging and Rejuvenation (WoSAR). – 2019. URL: https://doi.org/10.1109/ISSREW.2019.00078

9. Abdullah Z. H. Software Ageing Prevention from Software Maintenance Perspective – A Review / Z.H. Abdullah, J. H. Yahaya, Z. Mansor, A. Deraman // Journal of Telecommunication, Electronic and Computer Engineering. – 2017. – Volume 9(3-4). – P. 93-96.

10. Wu H. Software aging in mobile devices: Partial computation offloading as a solution / H. Wu, K. Wolter // IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). – 2015. – P. 125–131. URL: https://doi.org/10.1109/ISSREW.2015.7392057

11. Yakovyna V. S. User-Perceived Response Metrics in Android OS for Software Aging detection / V.S. Yakovyna, B. V. Uhrynovskyi // IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT). – 2020. URL: https://doi.org/10.1109/CSIT49958.2020.9322031

12. Test UI Performance – Android Developers. URL: https://developer.android.com/training/testing/performance

13. Memory allocation among processes – Android Developers. URL: https://developer.android.com/topic/performance/memory-management

14. Garg S. A Methodology for Detection and Estimation of Software Aging / S. Garg, A. V. Moorsel, K. Vaidyanathan, K. S. Trivedi // Proc. of the 9th Intl. Symp. on Software Reliability Engineering (ISSRE). – 1998.

References

1. Parnas D. L. Software aging / D. L. Parnas // Proceedings of 16th International Conference on Software Engineering. – 1994. – P. 279-287. URL: https://doi.org/10.1109/ICSE.1994.296790

2. Huang Y. Software rejuvenation: analysis, module and applications / Y. Huang, C. Kintala, N. Kolettis, N. Fulton // Proceedings of Twenty-Fifth International Symposium on Fault-Tolerant Computing. – 1995. – P. 381–390. URL: https://doi.org/10.1109/FTCS.1995.466961

3. Dohi T. Handbook of software aging and rejuvenation / T. Dohi, K. Trivedi, A. Avritzer // World Scientific Publishing Co Pte Ltd. – 2020. – P. 424. URL: https://doi.org/10.1142/11673

4. Grottke M. The fundamentals of software aging / M. Grottke, R. M. Jr, K. S. Trivedi // IEEE International Conference on Software Reliability Engineering Workshops. – 2008. – P. 1-6. URL: https://doi.org/10.1109/ISSREW.2008.5355512

5. Cotroneo D. Software aging analysis of the android mobile os / D. Cotroneo, F. Fucci, A. K. Iannillo, R. Natella, R. Pietrantuono // IEEE 27th International Symposium on Software Reliability Engineering. – 2016. – P. 478-489. URL: https://doi.org/10.1109/ISSRE.2016.25

6. Cotroneo D. A Comprehensive Study on Software Aging across Android Versions and Vendors / D. Cotroneo, A. K. Iannillo, R. Natella, R. Pietrantuono // Empirical Software Engineering. – 2020. – Volume 25(3). - P. 3357-3395.

7. Yakovyna V. S. Starinnia prohramnoho zabezpechennia mobilnykh dodatkiv: analiz problematyky / V.S. Yakovyna, B. V. Uhrynovskyi // Naukovyi visnyk NLTU Ukrainy. – № 30(2). – S. 107–112. URL: https://doi.org/10.36930/40300219

8. Cotroneo D. A Configurable Software Aging Detection and Rejuvenation Agent for Android / Cotroneo D., Simone, L. D., Natella, R., Pietrantuono, R., Russo, S. // 11th Intl Workshop on Software Aging and Rejuvenation (WoSAR). – 2019. URL: https://doi.org/10.1109/ISSREW.2019.00078

9. Abdullah Z. H. Software Ageing Prevention from Software Maintenance Perspective – A Review / Z.H. Abdullah, J. H. Yahaya, Z. Mansor, A. Deraman // Journal of Telecommunication, Electronic and Computer Engineering. – 2017. – Volume 9(3-4). – P. 93-96.

10. Wu H. Software aging in mobile devices: Partial computation offloading as a solution / H. Wu, K. Wolter // IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). – 2015. – P. 125–131. URL: https://doi.org/10.1109/ISSREW.2015.7392057

11. Yakovyna V. S. User-Perceived Response Metrics in Android OS for Software Aging detection / V.S. Yakovyna, B. V. Uhrynovskyi // IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT). – 2020. URL: https://doi.org/10.1109/CSIT49958.2020.9322031

12. Test UI Performance – Android Developers. URL: https://developer.android.com/training/testing/performance

13. Memory allocation among processes – Android Developers. URL: https://developer.android.com/topic/performance/memory-management

14. Garg S. A Methodology for Detection and Estimation of Software Aging / S. Garg, A. V. Moorsel, K. Vaidyanathan, K. S. Trivedi // Proc. of the 9th Intl. Symp. on Software Reliability Engineering (ISSRE). – 1998

ЯКОВИНА В. С. ORCID ID: 0000-0003-0133-8591 vitaliy.s.yakovyna@lpnu.ua
УГРИНОВСЬКИЙ Б. В. ORCID ID: 0000-0002-4356-192X bohdan.v.uhrynovskyi@lpnu.ua

Надійшла/Paper received : 23.03.2021 р. Надрукована/Printed : 02.06.2021 р.