

ІНТЕГРАЛЬНИЙ КРИТЕРІЙ ОЦІНКИ МЕТОДІВ ТЕСТУВАННЯ ДОДАТКІВ ДЛЯ МОБІЛЬНИХ ПРИСТРОЇВ

У статті запропоновано підхід до визначення інтегрального критерію оцінки методів тестування додатків для мобільних пристроїв.

Проведений аналіз процесу розробки додатків для мобільних пристроїв дозволяє виділити наступні особливості: процес розробки розглянутих додатків швидкоплинний, загальна тривалість проектів по впровадженню програми - не більше одного року; розробка відбувається по ітеративній схемі, тому вимагає оптимізації інтегрованих процесів функціонального тестування по часовому параметру; розробка високорівневого прототипу додатка з подальшою генерацією тестових сценаріїв, імовірно, дозволить оптимізувати процес впровадження додатків. В ітеративній схемі, в процесі розробки додатків для мобільних пристроїв, додаток надходить на тестування на системній стадії, минаючи процеси модульного та інтеграційного тестування. Таким чином, процес функціонального тестування зводиться до перевірки функціональності програми на рівні призначеного для користувача інтерфейсу. Критерій ефективності моделі генерації тестів в рамках загального процесу розробки програмного забезпечення дозволяє оцінити можливість задавати очікувані параметри станів розширеного кінцевого автомата, що подаються на вхід алгоритму генерації. За заданим значенням алгоритм зможе згенерувати кроки, призначені для перевірки відповідності поточного стану програми очікуваному стану, відповідно заданим параметрам. Виходячи з проведеного дослідження, можна зробити наступні висновки щодо методів тестування додатка для мобільних пристроїв: ефективність методу тестування додатка для мобільних пристроїв безпосередньо залежить від кількості ітерацій розробки, тому оптимізація методів тестування за часом зводиться до автоматизації процесів створення тестових сценаріїв і автоматизації їх виконання; найбільшу ефективність надає метод тестування на основі прототипів, при цьому тестові сценарії для тестування додатка для мобільних пристроїв генеруються з побудованого прототипу; необхідно мати можливість генерувати тестові сценарії таким чином, щоб використовувати їх в програмному методі автоматизації тестування з найменшими доопрацюваннями.

Ключові слова: інтегральний критерій, мобільні пристрої, моделі, методи тестування, генерації тестів, метод, алгоритм, тестові сценарії, прототип.

Вступ. Процес розробки додатків для мобільних пристроїв являє собою ітеративну схему, її застосування доцільно в проектах розробки додатка для мобільних пристроїв загальною тривалістю до одного року. Особливість ітеративної схеми полягає в прискореному просуванні до готового продукту внаслідок кількох ітерацій повного циклу розробки (рис. 1).

В результаті визначення функціональних вимог починається цикл з декількох етапів розробки готових альфа-версій продукту. На кожному етапі відбувається повна збірка додатка, системне тестування нового функціоналу і етап регресійного тестування. Після кількох ітерацій приймається рішення, що подальше доопрацювання додатка не є доцільним і відбувається вихід з циклу з додатком, готовим до випуску. Процес визначення вимог вимагає попереднього аналізу цільової аудиторії і ринку аналогічної продукції. Можливі кілька варіантів визначення бізнес-вимог до продукту: «мозковий штурм» за участю всіх залучених до процесу розробки людей; зустрічі із замовником і формування високорівневих вимог. Основна особливість отриманих вимог - високий рівень абстракції. Вимоги повинні бути зрозумілі будь-якій людині, залученій в процес розробки, і не повинні залежати від конкретної технології розробки. Існує кілька способів подання і зберігання бізнес-вимог:

- набір шаблонів - паперових або електронних. Кожен шаблон описує певну частину програми. Шаблони статичні і ніяк не взаємодіють між собою. Очевидні плюси шаблонів - їх

швидка розробка і наочність. Шаблони можна розробити таким чином, щоб вони дуже точно описували призначений для користувача інтерфейс продукту, що розробляється;

- модель додатка, написана на формальній мові. Для розробки формальних моделей необхідне знання відповідних інструментів моделювання, наприклад, UniTESK. Подібні формальні моделі дозволяють точно описувати внутрішню структуру програми, наприклад, ООП класи. Також формальні моделі дозволяють генерувати функціональні специфікації та тестові сценарії на основі побудованих моделей. Тестові сценарії, що генеруються, можуть застосовуватися для тестування методом «білого ящика», оскільки формальна модель описує внутрішню структуру програми. Такі моделі не дозволяють «побачити» готовий додаток, оскільки описують тільки функціонал, але не призначений для користувача інтерфейс;



Рис. 1. Ітеративна схема розробки додатка для мобільних пристроїв

- модель додатка, створена за допомогою спеціальних інструментів прототипування інтерфейса для користувача. Прототипи дозволяють точно описати поведінку додатка на системному рівні, тобто інтерфейс для користувача. При цьому прототипи ніяк не описують внутрішню структуру програми. Прототипи дозволяють генерувати функціональні специфікації та тестові сценарії, придатні для тестування методом «чорного ящика».

Постановка задачі. Проведений аналіз процесу розробки додатка для мобільних пристроїв дозволяє виділити наступні особливості: процес розробки розглянутих додатків швидкоплинний, загальна тривалість проектів по впровадженню програми - не більше одного року; розробка відбувається по ітеративній схемі, тому вимагає оптимізації інтегрованих процесів функціонального тестування за часовим параметром; розробка високорівневого прототипу додатка з подальшою генерацією тестових сценаріїв, імовірно, дозволить оптимізувати процес впровадження додатків.

Основна частина. В ітеративній схемі, в процесі розробки додатків для мобільних пристроїв, додаток надходить на тестування на системній стадії, минаючи процеси модульного і інтеграційного тестування. Таким чином, процес функціонального тестування зводиться до перевірки функціональності програми на рівні призначеного для користувача інтерфейсу. У загальному випадку взаємодія користувача з додатком відбувається за наступною схемою:

1. Користувач бачить на екрані мобільного пристрою деякий «вид» додатка. Цей вид містить елементи призначеного для користувача інтерфейсу (кнопки, поля введення і ін.), які дозволяють здійснювати різні дії (запити користувача).

2. Запит, що генерується користувачем, надходить в «логічну» частину додатка, яка обробляє запит і, можливо, звертається до бази даних за необхідними даними.

3. Отримавши дані, «логічна» частина генерує наступний «вид», який побачить користувач - як результат свого запиту.

4. Далі процес повторюється.

В схемі, зображеній на рис. 2, на кожному етапі розробки можна вибрати один із способів переходу до наступного етапу. Залежно від обраного переходу змінюється час досягнення наступного етапу. Сумарний час розробки залежить від шляху проходження зі стану 1 схеми в стан 7 і від кількості ітерацій циклу 1-7. Зі стану 1 в стан 7 можна прийти різними шляхами, але в кожному стані вибір наступного маршруту залежить від попереднього стану процесу розробки. Кожен з можливих маршрутів визначає метод тестування програми. Розглянемо найбільш поширені шляхи переходу із 1 в 7.



Рис. 2. Детальна ітеративна схема розробки додатка для мобільних пристроїв

Ручне тестування з документації: визначення вимог; розробка шаблонів; ручна розробка текстової функціональної специфікації; ручна розробка тестових сценаріїв; ручне тестування. Метод ручної розробки документів і тестів (*PP* метод) дуже гнучкий і дозволяє досягти будь-якого рівня тестового покриття. Однак цей метод вимагає великих часових витрат.

Автоматизація ручних тестів: визначення вимог; розробка шаблонів; ручна розробка текстової функціональної специфікації; ручна розробка тестових сценаріїв в текстовому вигляді; автоматизація тестових сценаріїв; автоматизоване тестування. Метод автоматичної розробки тестів (*PA* метод) вимагає текстового опису тестових сценаріїв. По суті, створюються тестові сценарії в текстовому вигляді, як і в випадку *PP* методу, а на основі даних тестових сценаріїв розробляються автоматизовані тести. Такий підхід вимагає значно більше часу на розробку тестів, але значно скорочує час проведення тестування. Отже, такий підхід ефективний при збільшенні кількості циклів розробки.

Тестування на основі формальної моделі (*ФМ* метод): визначення вимог; розробка формальної моделі; генерація специфікації; генерація тестових сценаріїв; автоматизоване тестування. Даний метод підлягає автоматизації, але не досить гнучкий в області розробки додатків для мобільних пристроїв, оскільки за допомогою формальних моделей важко описувати призначені для користувача інтерфейси, отже, важко отримати необхідне тестове покриття.

Тестування на основі прототипу: визначення вимог; розробка прототипу; генерація специфікації; генерація тестових сценаріїв; ручне та автоматизоване тестування. Метод розробки на основі прототипу (ППР метод) дозволяє добре описувати призначені для користувача інтерфейси, отже, отримати необхідне покриття. Метод також дозволяє генерувати різні типи тестових сценаріїв при наявності відповідних інструментів генерації і особливі правила побудови прототипів, що надає додаткову гнучкість в проведенні тестування.

Для мобільних додатків існують різні інструменти автоматизації тестування. У загальному випадку можна виділити два основні підходи до автоматизації тестів.

Програмний метод автоматизації. Метод, який передбачає використання певних бібліотек деякої мови програмування. Автоматизація тестів зводиться до написання допоміжних модулів тестування і написання тестових скриптів на мові програмування. Основний мінус цього підходу - значні часові витрати на реалізацію тестових скриптів.

Метод автоматизації з використанням playback інструментів. Такий підхід вимагає мінімальних навичок в програмуванні. Автоматизація тесту відбувається за допомогою проведення тестового сценарію на цільовому пристрої з пропущенням впливів через спеціальну програму (проксі-сервер), що записує вплив і дані. Після запису ця програма представляє отриману інформацію у вигляді автоматичного тесту. При такому підході автоматизація тесту зводиться до його виконання. Основний недолік цього підходу - будь-яка зміна в тестовому додатку вимагає перезапису тестових сценаріїв, пов'язаних зі зміненим функціоналом.

Інтегральний критерій оцінки методів тестування. Позначимо обраний шлях переходу по ітеративній схемі (рис. 2).

$$P_{12} - P_{23} - P_{34} - \dots = p_{ij} (ij \in [12, 23, 34, 56, 67]).$$

Позначимо сумарний час $T(p_{ij})$, а $N(p_{ij})$ - покриття, що відповідає N - перевіреним відгукам, відповідно до розробленої метрики тестування. Тоді оптимізація процесу тестування за двома параметрами задається наступним виразом:

$$\begin{cases} N(p_{ij}) \rightarrow \max \\ T(p_{ij}) \rightarrow \min \end{cases}, (ij \in [12, 23, 34, 56, 67]).$$

Оскільки N - деяке натуральне число, то цю умову можна записати в наступному вигляді:

$$T = \frac{\sum_{ij \in [12, 23, 34, 56, 67]} t_{ij}}{N} \rightarrow \min, \quad (1)$$

де t_{ij} - час, витрачений на даному етапі, в залежності від обраного способу проходження етапу, N - покриття відгуків додатка, які необхідно забезпечити. Даний вираз визначає інтегральний критерій ефективності підходу до тестування додатка для мобільних пристроїв в контексті всього процесу розробки та тестування.

Розглянемо кожен доданок в чисельнику окремо. Попередньо введемо деякий абстрактний час t , відповідний часу опису одного елемента користувацького інтерфейсу додатка. Очевидно, що всі складові часу у формулі (1), крім t_{56} , лінійно залежать від t , а часом t_{45} , який відповідає передачі збірки в тестування, можна знехтувати. Оскільки інтеграція засобу автоматизованого тестування в процес є одиничною операцією, цей час можна не враховувати при оцінці ефективності тестування. Введемо наступні коефіцієнти, що враховують залежності часу на кожному етапі розробки від часу t :

$C_{р\text{мод}}$ - коефіцієнт розробки прототипу - відносний час розробки прототипу в розрахунку на один елемент користувацького інтерфейсу,

$C_{оспец}$ - коефіцієнт отримання специфікації - відносний час розробки функціональної специфікації в розрахунку на один елемент користувацького інтерфейсу,

$C_{имп}$ - коефіцієнт імплементації, представляє відносний час розробки одного елемента користувацького інтерфейсу,

$C_{отест}$ - коефіцієнт отримання тестових сценаріїв, відносний час розробки тестових сценаріїв для перевірки одного елемента користувацького інтерфейсу,

$C_{тест}$ - коефіцієнт проведення тестування, відносний час проведення тестових сценаріїв для перевірки одного елемента користувацького інтерфейсу.

Нехай в ході розробки було пройдено K циклів розробки, на кожному з яких додавалося $\partial N_i (i=1...K)$ нових відгуків, які потрібно перевірити для забезпечення повного покриття.

Тоді, формула (1) прийме вид:

$$T = \frac{t(C_{р.мод} \cdot N + C_{оспец} \cdot N + C_{имп} + C_{отест} \cdot N + C_{тест} \cdot N)}{N} + \sum_{i=1}^K \frac{t\left(C_{р.мод} \cdot \partial N_i + C_{оспец} \cdot \partial N_i + C_{имп} \cdot \frac{\partial N_i}{N} + C_{отест} \cdot \partial N_i + C_{тест} \cdot N\right)}{\partial N_i}.$$

Тут слід зазначити, що при проходженні $i - \text{ї}$ ітерації циклу розробки:

1. Час імплементації доданих вимог в $\frac{\partial N_i}{N}$ раз менше часу імплементації повного функціоналу.

2. Застосовується регресійне тестування всього функціоналу, а не тільки доданих вимог. Для оцінки можна вважати, що середньою кількістю доданих вимог можна знехтувати в

порівнянні із загальною кількістю $\frac{\partial N_i}{N} \approx 0$ вимог. Отже, завдання оптимізації зводиться до мінімізації наступного виразу:

$$T = t \left[(C_{прог} + C_{оцін}) \cdot (K + 1) + (C_{р.мод} + C_{оспец} + C_{отест}) \cdot \left(1 + \frac{K \cdot \overline{\partial N}}{N} \right) + C_{имп} \frac{N + K \cdot \overline{\partial N}}{N^2} \right] \rightarrow . (2)$$

$\rightarrow \min$

Якщо розглянути кожен доданок окремо, видно, що для зменшення загального часу розробки, в першу чергу потрібно спробувати зменшити коефіцієнт виконання тестування $C_{тест}$, оскільки навпроти нього множник $(K + 1)$. У другу чергу потрібно спробувати зменшити суму коефіцієнтів $(C_{р.мод} + C_{оспец} + C_{отест})$.

Також слід зазначити, що в разі автоматизації тестових сценаріїв коефіцієнт $C_{отест}$ повинен враховувати час розробки автоматизованих тестів, необхідних для тестування одного відгуку додатка, а $C_{тест}$ розбивається на час прогону тестів і час оцінки результатів тестування: $C_{тест} = C_{прог} + C_{оцін}$. Формула (2) дає кількісну оцінку часу, витраченого на K циклів розробки і тестування додатка для мобільних пристроїв з покриттям N функціональних вимог. При розрахунку ефективності процесів тестування доданок $C_{имп} \frac{N + K \cdot \overline{\partial N}}{N^2}$ формули

(2) можна не враховувати, оскільки він залежить тільки від процесу розробки програмного коду і ніяк не пов'язаний з проведенням тестування. Введемо наступне поняття ефективності

методу тестування: $Eff = \frac{T_{PP}}{T_X}$, де T - час методу PP , T_X - час тестування по методу X . Слід

зазначити, що при такому розрахунку, ефективність PP методу дорівнює 1. Провівши більш

глибокий аналіз розглянутих методів, були виявлені допущення щодо параметрів, які можна зробити для оцінки ефективності методів.

З урахуванням прийнятих припущень ефективність розглянутих методів зводиться до наступних формул:

$$\left\{ \begin{array}{l} Eff_{PA(програми)}(K) = \frac{2.3 \cdot K^2 + 5 \cdot K}{0.8 \cdot K^2 + 4.5 \cdot K + 10} \\ Eff_{PA(playback)}(K) = \frac{2.3 \cdot K + 5}{K + 5.5} \\ Eff_{IP}(K) = \frac{2.3 \cdot K + 5}{0.8 \cdot K + 3.5} \\ Eff_{FM}(K) = \frac{2.3 \cdot K + 5}{1.6 \cdot K + 11.5} \end{array} \right. . \quad (3)$$

Виходячи з результатів формалізації ітеративної схеми розробки, можна зробити припущення про ефективність прототипного підходу в розробці/тестуванні додатка для мобільних пристроїв.

Частковий критерій ефективності моделі генерації. Основою застосування прототипного підходу в тестуванні є ефективний алгоритм генерації тестових сценаріїв. Згенеровані тестові сценарії можуть застосовуватися як для ручного тестування, так і для автоматизованого тестування з використанням програмного підходу автоматизації. Пропонується ввести такі критерії ефективності моделі генерації тестів в рамках загального процесу розробки програмного забезпечення: середня ефективність алгоритму генерації; середня швидкість роботи алгоритму генерації; середнє забезпечуюче тестове покриття; можливість генерації кроків на перевірку результатів тестування; розширюваний формат вихідних даних. Усереднення береться за різними вхідними додатків, що піддаються тестуванню.

Середня ефективність алгоритму генерації. Щоб оцінити ефективність алгоритму генерації, пропонується використовувати наступний підхід: нехай N - кількість тестових сценаріїв, згенерованих алгоритмом, L - середня довжина тестового сценарію. Процес оцінки ефективності алгоритму можна представити наступною схемою: обидва параметри залежать від наданого графа розширеного кінцевого автомата. Відповідно до теореми Ейлера, у зв'язковому наведеному графі існує єйлерів цикл тоді і тільки тоді, коли півстепінь результату дорівнює півстепені заходу. Тобто кількість вхідних у вузол гілок дорівнює кількості вихідних. Використовуючи цю теорему, можна зробити оцінку нижньої межі кількості переходів, необхідних для покриття всіх переходів розширеного кінцевого автомата.

$$N_e = \sum_{s_i \in S} \max(s_i^{in}, s_i^{out}).$$

Відповідно до визначення розширеного кінцевого автомата (РКА) прототипу, відповідний граф РКА є зв'язковим, оскільки в РКА додається новий стан, тільки якщо існує перехід з поточного стану в новий. Для «ейлеризації» графа необхідно обійти всі його вузли і «доповнити» стан мінімальною кількістю вихідних або вхідних гілок, таким чином, щоб кількість вхідних гілок в даний вузол дорівнювала кількості вихідних гілок. Очевидно, після подібного доповнення кількість вихідних гілок графу стану st дорівнюватиме кількості вхідних гілок і дорівнюватиме $\max(s_i^{in}, s_i^{out})$. Таким чином, загальна кількість переходів, яку потрібно зробити для обходу всього «ейлеризованого» графа буде $\sum_{s_i \in S} \max(s_i^{in}, s_i^{out})$.

Тоді показник ефективності числа згенерованих кроків наступний:

$$Eff(G) = \frac{N_e}{N \times L} \cdot 100\% . \quad (4)$$

де G - граф прототипу додатка для мобільних пристроїв, $N \times L$ - загальна кількість переходів, здійснених алгоритмом в наслідок обходу, $N_e = \sum_{s_i \in S} \max(s_i^{in}, s_i^{out})$ - мінімальна і

достатня кількість переходів, які потрібно здійснити для покриття всіх переходів РКА (відповідно до теореми Ейлера).

У загальному випадку можлива ситуація, коли пропонуваній алгоритм генерації не в змозі забезпечити повне тестове покриття через попадання в «нескінченний» цикл, або при попаданні в тупиковий стан. У цьому випадку даний алгоритм забезпечить покриття менше 100%. Пропонуваній критерій усереднює забезпечуванім алгоритмом покриття за різними РКА додатками, що подаються на вхід.

Можливість генерації кроків на перевірку результатів тестування. Цей критерій дозволяє оцінити можливість задавати очікувані параметри станів РКА, що подаються на вхід алгоритму генерації. За заданим значенням алгоритм зможе згенерувати кроки, призначені для перевірки відповідності поточного стану програми очікуваному стану, що відповідає заданим параметрам.

Висновки. Виходячи з проведеного дослідження, можна зробити наступні висновки щодо методів тестування додатка для мобільних пристроїв:

- загальна задача оптимізації методів тестування додатка для мобільних пристроїв

описується формулою
$$T = \frac{\sum_{ij \in [12,23,34,56,67]} t_{ij}}{N} \rightarrow \min ;$$

- ефективність методу тестування додатка для мобільних пристроїв безпосередньо залежить від кількості ітерацій розробки, тому оптимізація методів тестування за часом зводиться до автоматизації процесів створення тестових сценаріїв і автоматизації їх виконання;

- найбільшу ефективність надає метод тестування на основі прототипів, при цьому тестові сценарії для тестування додатка для мобільних пристроїв генеруються з побудованого прототипу;

- необхідно мати можливість генерувати тестові сценарії таким чином, щоб використовувати їх в програмному методі автоматизації тестування з найменшими доопрацюваннями;

- необхідно розробити модель алгоритму генерації тестових сценаріїв, оптимізовану за частинним критерієм ефективності, що описується формулою: $Eff(G) = \frac{N_e}{N \times L} \cdot 100\% .$

ЛІТЕРАТУРА:

1. Ханссон Д. Х. Гибкая разработка веб-приложений в среде Rails / Д. Х. Ханссон, Д. Томас – Санкт-Петербург – Питер, 2008. – 720с.
2. Салмре Иво. Конечный автомат для пользовательского интерфейса. Программирование мобильных устройств на платформе .NET Compact Framework / Иво Салмре – Изд-во: Издательский дом "Вильямс", 2006. – 736с.
3. Быстрая разработка программ. Принципы, примеры, практика /Роберт Мартин, Джеймс Ньюкирк, Роберт Косс – Изд-во: Диалектика-Вильямс, 2004. – 752 с.
4. Rails 4. Гибкая разработка веб-приложений / Сэм Руби, Дэйв Томас, Дэвид Ханссон – Изд-во: Питер, 2014. – 448 с.
5. Гибкая разработка программ на Java и C++. Принципы, паттерны и методики. /Роберт С. Мартин, Джеймс Ньюкирк, Роберт Косс – Изд-во: Диалектика-Вильямс, 2016. – 704 с.
6. Гарднер Л. Разработка веб-сайтов для мобильных устройств / Л. Гарднер, Д.Григсби. – Питер – Москва, 2013. – 448 с.
7. Дакетт Джон HTML и CSS. Разработка и дизайн веб-сайтов (+ CD-ROM) / Джон Дакетт – Эксмо - Москва, 2013. – 480 с.

8. Китинг Джоди Flash MX. Искусство создания web-сайтов / Джоди Китинг –ТИД ДС – Москва, 2012. - 848 с.
9. Кузнецов М. PHP. Практика создания Web-сайтов / М. Кузнецов, И.Симдянов–БХВ-Петербург - Москва, 2012. – 347 с.
10. Митчелл Скотт 5 проектов Web-сайтов от фотоальбома до магазина / Скотт Митчелл –М.: НТ Пресс - Москва, 2013. – 224 с.
11. Фрейен Бен HTML5 и CSS3.Разработка сайтов для любых браузеров и устройств / Бен Фрейен– Питер - Москва, 2014. - 304 с.
12. Чебыкин Ростислав Разработка и оформление текстового содержания сайтов / Ростислав Чебыкин – БХВ-Петербург – Москва, 2014. – 528 с.
13. SEO. Искусство раскрутки сайтов / Эрик Энж, Стефан Спенсер, Рэнд Фишкин, Джесси Стрикчиола –БХВ-Петербург – Москва, 2014. – 668 с.

REFERENCES:

1. Hansson, D. X. and Tomas, D. (2008),” Gibkaya razrabotka veb-prilozheniy v srede Rails” [Flexible development of web applications in the Rails environment], Sankt-Peterbur - Piter, 720p.
2. Salmre, Ivo (2006),” Konechniy avtomat dlya polzovatel'skogo interfeysa. Programirovanie mobilnykh ustroystv na platforme .NET Compact Framework” [A state machine for the user interface. Programming mobile devices on the .NET Compact Framework], Izd-vo: Izdatelskiy dom "Vilyams", 736p.
3. Martin, Robert, Nyukirk, Dzheymys and Koss, Robert (2004),” Bystraya razrabotka programm. Printsipy, primery, praktika” [Rapid development of programs. Principles, examples, practice], Izd-vo: Dialektika-Vilyams, 752 p.
4. Rubi, Sem, Tomas, Deyv and Hansson, Devid (2014),” Rails 4. Gibkaya razrabotka veb-prilozheniy” [Rails 4. Flexible Web Application Development], Izd-vo: Piter, 448p.
5. Martin, Robert, Nyukirk, Dzheymys and Koss, Robert (2016),” Gibkaya razrabotka programm na Java i C. Printsipy, patterny i metody.” [Flexible development of Java and C ++ programs. Principles, Patterns and Techniques], Izd-vo: Dialektika-Vilyams, 704p.
6. Gardner, L. and Grigsby, D. (2013),” Razrabotka veb-saytov dlya mobilnykh ustroystv” [Development of websites for mobile devices], Piter - Moskva, 448p.
7. Dakett, Dzhon (2013),” HTML i CSS. Razrabotka i dizayn veb-saytov (CD-ROM)” [HTML and CSS. Development and design of websites (+ CD-ROM)], Eksmo - Moskva, 480p.
8. Kiting, Dzhodi (2012)” Flash MX. Iskustvo sozdaniya web-saytov” [Flash MX. The Art of Creating Websites], TID DS - Moskva, 848p.
9. Kuznetsov, M. and Simdyanov, I. (2012),” PHP. Praktika sozdaniya Web-saytov” [PHP. The practice of creating Web sites], BHV-Peterburg - Moskva, 347p.
10. Mitchell, Skott (2013), ”5 proektov Web-saytov ot fotoalboma do magazina” [5 projects of Web sites from photo album to shop], M.: NT Press - Moskva, 224p.
11. Freyen, Ben (2014),” HTML5 i CSS3.Razrabotka saytov dlya lyubyykh brauzerov i ustroystv” [HTML5 and CSS3. Development of sites for all browsers and devices], Piter - Moskva, 304p.
12. Chebyikin, Rostislav (2014),” Razrabotka i oformlenie tekstovogo soderzhaniya saytov” [Development and design of text content of sites], BHV-Peterburg - Moskva, 528p.
13. Enzh, Erik, Spenser, Stefan, Fishkin, Rend and Strikchiola, Dzhesi (2014),” SEO. Iskustvo raskrutki saytov” [SEO. The art of website promotion], BHV-Peterburg - Moskva, 668p.

к.т.н., доц. Браун В.О., к.т.н. Джулий В.Н., Кызыма К.А., Селюков Д.А., Солодеева Л.В.
ИНТЕГРАЛЬНЫЙ КРИТЕРИЙ ОЦЕНКИ МЕТОДОВ ТЕСТИРОВАНИЕ
ПРИЛОЖЕНИЙ ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ

В статье предложен подход к определению интегрального критерия оценки методов тестирования приложений для мобильных устройств. Проведенный анализ процесса разработки приложений для мобильных устройств позволяет выделить следующие особенности: процесс разработки рассмотренных приложений быстротечный, общая продолжительность проектов по внедрению программы - не более одного года; разработка происходит по итеративной схеме, поэтому требует оптимизации интегрированных процессов функционального тестирования по временному параметру; разработка высокоуровневого прототипа приложения с последующей генерацией тестовых сценариев, вероятно, позволит оптимизировать процесс внедрения

приложений. В итеративной схеме, в процессе разработки приложений для мобильных устройств, приложение поступает на тестирование на системной стадии, минуя процессы модульного и интеграционного тестирования. Таким образом, процесс функционального тестирования сводится к проверке функциональности программы на уровне пользовательского интерфейса. Критерий эффективности модели генерации тестов в рамках общего процесса разработки программного обеспечения позволяет оценить возможность задавать ожидаемые параметры состояний расширенного конечного автомата, которые подаются на вход алгоритма генерации. По заданным значениям алгоритм сможет сгенерировать шаги, предназначенные для проверки соответствия текущего состояния программы ожидаемому состоянию, в соответствии заданным параметрам. Исходя из проведенного исследования, можно сделать следующие выводы относительно методов тестирования приложения для мобильных устройств: эффективность метода тестирования приложения для мобильных устройств напрямую зависит от количества итераций разработки, поэтому оптимизация методов тестирования по времени сводится к автоматизации процессов создания тестовых сценариев и автоматизации их выполнения; наибольшую эффективность дает метод тестирования на основе прототипов, при этом тестовые сценарии для тестирования приложения для мобильных устройств генерируются из построенного прототипа; необходимо иметь возможность генерировать тестовые сценарии таким образом, чтобы использовать их в программном методе автоматизации тестирования с наименьшими доработками.

Ключевые слова: интегральный критерий, мобильные устройства, модели, методы тестирования, генерации тестов, метод, алгоритм, тестовые сценарии, прототип.

**Ph.D. Braun V.O., Ph.D. Dzhulij V.M., Kyzyma K.A., Selyukov D.O., Solodeeva L.V.
INTEGRAL CRITERIA FOR EVALUATION OF APPLICATION TEST
METHODS FOR MOBILE DEVICES**

The article describes an approach of an integral criteria definition for assessing application testing methods for mobile devices. The analysis of mobile application development process allows to separate the following features: the development of the considered applications is brief process, the project lifetime considering the implementation of application is not more than one year; the development is an iterative process, therefore it requires optimization of the integrated processes of functional testing based on the time parameter; the development of a high-level prototype application with the subsequent generation of test scenarios is about optimizing the application implementation process. In the iterative scheme, during the applications developed for mobile devices, the application enters testing at the system stage, passing the processes of modular and integration testing. Thus, the process of functional testing is to verify the functionality of the program at the level of the user interface. The criteria for the effectiveness of the test generation model within the overall software development process is to evaluate the ability to specify the expected parameters of the states of the advanced finite automaton submitted to the input of the generation algorithm. The algorithm will be able to generate steps designed to verify that the current state of the program is in the expected state according to the specified parameters. Based on the study, we can draw the following conclusions about mobile application testing methods: the effectiveness of the application testing method for mobile devices depends on the number of iterations of the development; therefore, the optimization of testing methods over time is to automate the process of test scenarios creation and automate their execution; the prototype testing method is most effective with test scenarios for mobile application testing being generated from the prototype built; you need to be able to generate test scenarios in such a way as to use them in the software test automation method with the least adjustments.

Keywords: integral criteria, mobile devices, models, testing methods, test generation, method, algorithm, test scenarios, prototype.