УДК 004.415.5

ЯКОВЛЄВА С.О., ст.викладач
МІХАЙЛУЦА О.М., к.т.н., доцент
ПОЖУЄВ А.В., к.ф.-м.н, доцент

Запорізька державна інженерна академія

ДОСЛІДЖЕННЯ ВПЛИВУ ФУНКЦІОНАЛЬНИХ ВИМОГ НА ЯКІСТЬ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Вступ. Успіх процесу розробки сучасних програмних систем залежить від багатьох факторів. Для створення якісного програмного забезпечення, яке б повністю задовольняло потреби користувача, необхідно власне виявити проблеми клієнтів та забезпечити їх кваліфіковане вирішення.

Основним наслідком проблем з вимогами є переробка вже існуючих частин програмного продукту, на що, згідно з сучасними дослідженнями, витрачається від 30 до 50% загального бюджету розробки [1]. Неякісно протестована система може принести до 70% збитків компанії, при цьому ризики значно зростають на завершальних етапах розробки та на стадіях впровадження та супроводу [2].

Необхідно впроваджувати в життєвий цикл розробки програмного продукту не лише систематичний підхід до виявлення, організації та документування вимог до системи, а й процес керування вимогами, в ході якого виробляється та забезпечується згода між замовником та групою, що виконує проект щодо мінливих вимог до системи. При цьому постає проблема відстеження вимог для підтвердження якості реалізованої функціональності. Трасованість вимог документує залежності та логічні зв’язки окремих вимог та інших елементів програмного продукту. До цих елементів відносяться бізнес-правила, архітектура та компоненти дизайну, модулі вихідного коду, варіанти тестування та файли довідки користувачів. Інформація трасування полегшує виконання аналізу впливу, допомагаючи визначити всі робочі продукти, які може знадобитися змінити для реалізації запропонованої зміни вимоги.

В більшості випадків розуміння та інтерпретація вимог продовжує еволюціонувати в процесі проектування та розробки програмного забезпечення. Реакція команди-розробника на зміну вимог при цьому має бути неупередженою – необхідно розуміти неминучість змін та планувати кроки по зменшенню проблем, пов’язаних зі змінами. Спроба "заморозити" вимоги призведе до накопичування латентних необхідних змін, які потім вибухнуть перед командою-розробником і спричинять стрес та велику кількість переробок. Іншою крайністю даного процесу є необмежені можливості будь-якого члена команди змінювати вимоги до системи без відповідних погоджень, що може призвести до повного заплутування цілей проекту.

Постановка задачі. Сучасні методи розробки програмних систем припускають впровадження процесу тестування ПЗ водночас з процесом виявлення вимог та розробкою програмного продукту по специфікації, що надана аналітиками.

Коли готова перша порція вимог, тестувальник вже може приступати до написання тестових випадків або сценаріїв. Якщо вимоги неякісні – суперечливі, двозначні, неповні тощо – тестувальник виявить це в процесі написання тестів, тому що методики, які він використовує, направлені на покриття всієї множини станів і переходів, вхідних даних та ситуацій відмови. Таким чином, відбувається побудова моделі майбутньої системи за наявними вимогами та складаються тести для перевірки цієї моделі. Якщо модель описана не повністю, або її частки не узгоджені між собою, цю модель неможливо буде перевірити, тож написання тестів дає зворотний зв'язок про якість набору функціональних вимог. З іншого боку, правильно і точно описані бізнес-вимоги також збага-

чують тестування у вигляді інформації про пріоритети. Проведення трасування від функціональних вимог до вимог вищих рівнів – користувальницьких та бізнес-вимог – дозволяє перевірити коректність побудови специфікацій та простежити відображення вимог на компоненти архітектури системи. Спілкуючись з аналітиками, тестувальники краще розуміють, які функції або характеристики якості системи є найбільш важливими, що дає можливість розподіляти зусилля з тестування відповідно до бізнес-пріоритетів. Тестування та вимоги до системи пов'язані між собою синергетичними відносинами, оскільки представляють собою взаємодоповнюючі погляди на систему [1].

Результати роботи. Основою процесу роботи з вимогами є їх виявлення, результатом чого має бути узгоджене уявлення про потреби всіх зацікавлених в проекті осіб. зазвичай використовується мозковий штурм, інтерв'ювання та анкетування, розкадрування та обігравання ролей. Кожна вимога, виявлена в процесі спілкування з замовником, має повно відображати бажані характеристики майбутньої системи, не суперечити іншим вимогах та бути однозначно сприйнятою всіма учасниками проекту.

Дуже важливим при цьому є усвідомлення кожного члена команди-розробника своєї ролі в процесі створення програмного продукту та розуміння специфікації проекту в цілому для успішного виконання власних завдань. Вміння працювати в команді повинно бути підкріплено відповідальністю за кінцеві результати кожного з її членів, відповідною кваліфікацією та прагненням самовдосконалення.

Для формалізації функціональних вимог системи, які покращать розуміння вимог розробниками та тестувальниками, можна використовувати наступні методи:

- побудова варіантів використання;
- написання псевдокоду;
- скінчені автомати;
- графічні дерева рішень;
- моделі «сутність-зв'язок»;
- діаграми діяльності;
- таблиці рішень;
- об'єктно-орієнтовані моделі;
- схеми потоків даних [3].

При аналізі функціональних вимог основну увагу слід приділяти визначенню основних критеріїв успішності проведених тестів. Задані у вимогах характеристики будуть базовими навантажувальними точками тестованої системи. Всі одержувані результати порівнюватимуться з ними для прийняття рішення про завершення або проведення подальшого регресійного тестування.

Оскільки процеси тестування та розробки знаходяться на одному ребрі трикутника час-гроші-якість, необхідно жорстко зв'язувати на синхронізувати ці два напрямки, інакше існує великий ризик того, що здвиг плану розробки, який впливає на дату поставки версії в тестування, призведе до здвигу робіт по тестуванню, а це може стати причиною випуску неякісного програмного продукту. До того ж, вже на етапі видобуття вимог до системи необхідно чітко розставляти пріоритети вимог, які потім будуть перетворені на окремі функції, створюючи документ про образ та межі проекту, для з'ясування ризиків, які можуть виникнути при скороченні строків розробки або тестування.

Вимоги повинні використовуватися для оцінки масштабу проекту. Об'єм продукту можна визначити виходячи як з текстових вимог, так і моделей аналізу, прототипів та елементів користувальницького інтерфейсу. При переході від вимог до планів проекту необхідно враховувати не лише самі вимоги, функції та елементи керування, елементи графічного інтерфейсу, які необхідно реалізувати, а й кількість їх варіантів тестування. Також слід прийняти до уваги стабільність бізнес-цілей замовника та інформацію про продуктивність спеціалістів, які приймають участь у проекті, засновану на по-

передньому досвіді. Необхідно чітко дотримуватись межі та образу проекту, щоб уникнути його розростання, шляхом перенесення надлишкових можливостей до пізніших версій. Впровадження необхідні змін які можуть вплинути на базову функціональність продукту повинно проводитись на поточному рівні реалізації після урахування всіх можливих наслідків.

Коректна оцінка часу та зусиль які знадобляться для розробки програмного продукту дозволить зосередитись на всіх етапах життєвого циклу, включаючи верифікацію, яка зазвичай страждає при скороченні строків розробки, що значно підвищить цінність розроблюваної системи. Вимоги є одним з артефактів життєвого циклу програмного продукту, тож вони також мають підлягати верифікації. Ця процедура дозволить виявити нечіткі вимоги, або вимоги, які не можна перевірити, а також ті, що були визначені недостатньо чітко для розробки. Неякісно виявлені або сформульовані вимоги неминуче призведуть до неточних розрахунків розміру, витрат і графіку. На розробку вимог середнього проекту необхідно витрачати близько 7% всіх ресурсів та 38,6% часу [4].

Успішне виявлення функціональних вимог на початку роботи з проектом не забезпечує успіху проекту, адже вимоги мають властивість змінюватися під час розробки програмного продукту. Необхідно здійснювати керування вимогами, яке полягає в керуванні змінами базової версії вимог, підтримці планів проекту актуальними у відповідності до змін вимог, контролі стану вимог та керуванні логічними зв'язками між вимогами та їх втіленням у артефактах проекту.

Для цього необхідно створити базову версію вимог, тобто набір функціональних і нефункціональних вимог, які розробники зобов'язалися реалізувати в певній версії. Ця версія має бути донесена до всіх учасників проекту і бути поширена на загальному ресурсі з можливістю сповіщення всіх членів команди про внесені зміни. Крім того, встановлюється єдиний канал контролю змін, яких забезпечить негайне визначення впливу запропонованої зміни на систему для оцінки всіх можливих факторів ризику та відстеження трасованості зміни на всі артефакти проекту, включаючи верифікацію та тестування. Кожна вимога проекту, функціональна чи нефункціональна, повинна бути унікально позначена і всі втілені зміни вимог, що з нею пов’язані, задокументовані. Таким чином, для кожної вимоги встановлюється власний статус, який дозволяє оцінити не лише стан виконання окремої функціональності, а й стан розроблюваного проекту в цілому.

Кожну зміну вимог в процесі розробки програмного продукту має бути відстежено та протестовано. Можливості перегляду стану проекту повинні бути доступні не лише менеджерам, аналітикам та керівникам проекту, а й всім учасникам команди розробки та тестування для усвідомлення стану готовності розроблюваної системи. Microsoft Visual Studio надає спеціальні можливості як для тестування (Team Foundation Server Test Manager), так і для документування та керування змінами вимог (Team Foundation Server Requirements Management), які можуть бути використані під час процесу розробки сучасних програмних продуктів.

Висновки.Таким чином, розробка якісної специфікації вимог та її подальше використання для побудови тестів на основі потреб користувачів є основою вчасного випуску високоякісних програмних проектів. Всі члени команди проекту мають не лише усвідомлювати свою роль у процесі створення програмного продукту, а й наслідки від створення неузгодженої або надлишкової функціональності, яка не була зазначена у вимогах до проекту і може значно вплинути на об’єм, витрати та графік розроблюваного проекту.

Для покращення підтримки співпраці аналітиків та тестувальників рекомендується використання автоматизованих засобів з підтримкою бази вимог та можливістю

їх відстеження та трасування в архітектуру та артефакти системи та подальші приймальні випробування готового продукту.

ЛІТЕРАТУРА

1. Леффингуелл Д. Принципы работы с требованиями к программному обеспечению / Д.Леффингуелл, Д.Уидриг. – М.: Вильямс, 2002. – 448с.
2. Вигерс К. Разработка требований к программному обеспечению / К.Вигерс. – М.: Издательско-торговый дом «Русская Редакция», 2004. – 576с.
3. Тамре Л. Введение в тестирование программного обеспечения / Л.Тамре; пер. с англ. – М.: Вильямс, 2003. – 368с.
4. Мацяшек Л. Анализ требований и проектирование систем. Разработка информационных систем / Л.Мацяшек. – М.:Вильямс, 2002. – 428с.
5. Алистер Коберн. Современные методы описания функциональных требований к системам / Коберн А. – М.: Лори, 2002. – 288с.
6. Соммервилл И. Инженерия программного обеспечения / И.Соммервил; пер. с англ. – 6-е изд. – М.: Вильямс, 2002. – 624с.

Надійшла до редколегії 09.09.2014.

УДК 539.3

ШЕВЧЕНКО Ю.Н., академик НАН Украины
ГАЛИШИН А.З., д.т.н.
СТЕБЛЯНКО П.А.*, д-р физ.-мат. наук
БАНЯС М.В., к. физ.-мат. н.
ДЕГТЯРЕНКО П.Г.**
ТОНКОНОЖЕНКО А.М.**

Институт механики им. С.П.Тимошенко НАН Украины
*Днепродзержинский государственный технический университет
**Государственное предприятие «Конструкторское бюро «Южное»

ОПРЕДЕЛЕНИЕ НЕСТАЦИОНАРНЫХ ТЕМПЕРАТУРНЫХ ПОЛЕЙ В ТОНКИХ СЛОИСТЫХ ОБОЛОЧКАХ ВРАЩЕНИЯ ПРИ КОМБИНИРОВАННОМ ТЕПЛООБМЕНЕ С ОКРУЖАЮЩЕЙ СРЕДОЙ

Введение. В технике находят широкое применение тонкостенные конструкции, выполненные в виде тонкостенных слоистых оболочек вращения и работающие в условиях неизотермического нагружения. К ним относятся сопла ракетных двигателей, элементы стартового оборудования для запуска космических ракет, элементы паро- и газотурбинных установок, трубопроводов и др. При интенсивном нагреве в начальные моменты времени в этих конструкциях возникают большие градиенты температуры, которые могут вызвать возникновение необратимых деформаций, которые зачастую приводят к разрушению конструкции. Поэтому для оценки прочности таких конструкций необходимо наиболее точно определять возникающее в них распределение температуры. В настоящее время разработаны экспериментальные и численные методы определения нестационарных температурных полей в слоистых оболочках. Обзор работ, посвященных методам расчета слоистых оболочек при конвективном теплообмене с окружающей средой, приведен в статье [1], где также изложена методика решения осесимметричной задачи теплопроводности для тонких слоистых оболочек. В настоящей статье рассматривается задача об определении нестационарных осесимметричных температурных полей в тонких слоистых оболочках вращения при комбинированном