РОЗРОБКА, КЕРОВАНА ФУНКЦІЯМИ (FDD)

Васюта Василь Васильович
к.т.н., доцент
Барсуков Станіслав Георгійович
студент
Національний університет Полтавська політехніка
імені Юрія Кондратюка
м. Полтава, Україна

Вступ. /Introductions. FDD – це методологія, що широко розглядається як заснована на чотирьох цінностях Agile та дванадцяти принципах Agile, викладених у Agile Manifesto. Це була одна з шести методологій, представлених на зустрічі, які створили Agile Manifesto.

Мета роботи. /Alim. Розглянути сутність однієї із найпопулярніших методології та практик, використовуваних в сучасних процесах розробки програмного забезпечення – розробки, керованої функціями (FDD).

Матеріали та методи. /Materials and methods. Для досягнення поставленої мети у роботі використані методи теоретичного узагальнення та аналізу.

Результати та обговорення. /Results and discussion. Робота починається з побудови об'єктної моделі предметної області та визначення всіх функцій та наборів функцій (груп функцій). Після визначення пріоритетів функцій робота продовжується ітеративно та поступово [1].

Поєднання традиційного планування з практиками гнучкого складання допомагає FDD вписатися в традиційні корпоративні структури.

Розробка, заснована на функціях, включає п'ять кроків у два етапи: планування та будівництво (рис. 1).

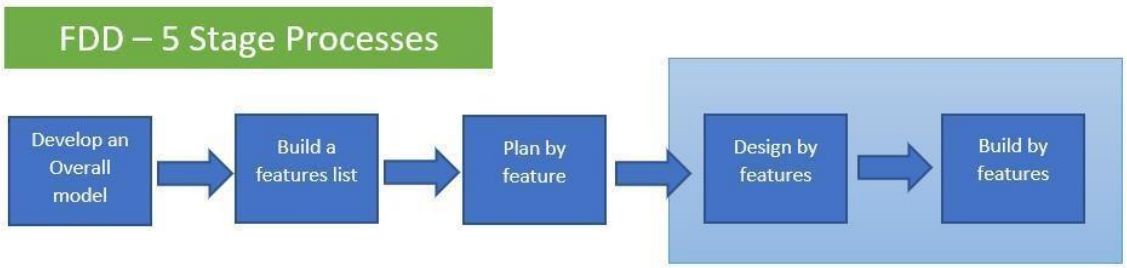

Рис. 1. Процес FDD

Етап планування полягає у досягненні загального розуміння обсягу продукту (нової програмної системи) між замовником та його користувачами, а також організацією-розробником. Таким чином FDD допомагає розпочати еволюцію, а не здійснювати революцію у напрямку гнучкої розробки.

Крок 1. Розробка вихідної моделі.

Першим кроком у FDD є малюнок вихідної об'єктної моделі предметної області. Команди розробників та експертів у предметній галузі (наприклад, майбутні користувачі в компанії клієнта) визначають модель, яка відображає їхнє загальне розуміння важливих об'єктів у новій системі. Головний архітектор підтримує та спрямовує їх у цій роботі.

Коли це буде зроблено всі моделі перевіряються, щоб запропонувати вихідну модель для наступних кроків. Це може бути одна з моделей або модель, яка складається з однієї або декількох моделей окремої команди.

Крок 2. Складання списку функцій.

Для цього ви розділите модель предметної області на піддомени, кожен із яких представляє бізнес-функцію. Потім для кожного піддомену ви перераховуєте всі функції, необхідні для продукту для підтримки клієнта в цій бізнес-функції. Тісно пов'язані функції часто групуються набори функцій. Набір функцій також може бути функцією, розбитою на дрібніші, тому що в іншому випадку для її завершення знадобилося б більше двох тижнів [2].

Крок 3. Планування за функціями.

На третьому етапі планують порядок розробки функцій та наборів функцій, враховуючи наступні чинники:

- які функції необхідні користувачам вашого клієнта найбільш

терміново;

- які функції принесуть найбільшу користь найближчим часом;
- оцінка часу розробки – не варто залишати найважливіші функції насамкінець;
- технічні залежності між функціями;
- ризики, пов'язані з функціями;
- доступні кадри, їхні навички та досвід;
- навантаження.

На етапі проектування та складання FDD ви працюєте зі списком функцій для кожної функції. Робота виконується двотижневими циклами або ітераціями і, таким чином, є ітеративною і поетапною. Суворий двотижневий цикл – ось чому FDD хоче, щоб ви розбили функції, щоб вони відповідали цьому.

У FDD можна формувати та реформувати команди з урахуванням вимог тих функцій, які ви будуватимете наступного разу. Як правило, підбір людей конкретної функціональної групи є прерогативою головного програміста.

Крок 4. Дизайн за функціями.

Кожна функціональна група працює над детальним дизайном функцій, призначених для поточної ітерації. Розробники та експерти в предметній галузі, яким при необхідності допомагає головний архітектор та головний програміст, використовують методи моделювання UML, такі як діаграми класів, діаграми послідовностей та діаграми переходу станів, щоб вказати, що робитиме функція і як вона це робитиме. Працюючи над цими конкретними проектами, ми розширюємо знання про продукт і повідомляємо про будь-які зміни, необхідні для вихідної об'єктної моделі предметної галузі.

Кожна ітерація в кінці етапу проектування, перевірка всією командою гарантує, що збіги будуть виявлені, і всі команди будуть обізнані про загальну модель та дизайн і залишаться в рамках вихідної концепції продукту.

Крок 5. Побудова за функціями.

Потім кожна функціональна група працює над тим, щоб перетворити свій проект на працююче програмне забезпечення, протестувати його та зібрати

відгуки експертів у предметній галузі, щоб переконатися, що функція працює належним чином. Коли все буде гаразд, вони поєднають свою роботу з усім, що було створено в цій та попередніх ітераціях.

Розробка, керована функціями, є поєднанням попереднього проектування вихідної моделі предметної області – і планування, а також ітеративної та інкрементної реалізації. Таким чином, цей метод орієнтований на архітектуру та є гнучким.

Він гнучкий у тому, що наголошує на задоволення потреб кінцевих користувачів, співпраця з експертами в предметній області клієнтів і розбиває вимоги на невеликі функції, покликані забезпечити цінність для бізнесу [3].

В іншому він набагато менш маневрений.

Об'єктно-орієнтований дизайн займає чільне місце у розробці, керованій функціями. Функції та набори функцій. Функції – це невеликі, цінні клієнта частини функціональності продукту. Вони виражаються через дію, результат та об'єкт [4].

Володіння індивідуальним класом (кодом). У кожного класу або пов'язаного набору функцій у коді є один власник, який відповідає за продуктивність та якість цього коду.

Функціональні групи у FDD динамічні, вони формуються та розформовуються з власників класів у міру необхідності для функцій, які будуть побудовані далі.

Інспекції. FDD використовує інспекції дизайну та інспекції коду для запобігання помилкам – або, швидше, для їх виявлення до того, як вони вислизнуть у дику природу.

Добре помітні звіти про хід роботи. FDD визначає вичерпний та зрозумілий набір звітів про хід виконання. Він використовує ряд етапів для послідовного звіту про хід виконання функцій та статуси кодів кольорів для зручного загального уявлення про хід виконання проекту. Два звіти про хід роботи, характерні для FDD – це діаграми паркувань і діаграми завершених функцій.

Висновки. /Conclusions. В цілому FDD – добрий крок у бік від водопадного підходу, але він не відповідає багатьом загальноприйнятим методам гнучкої розробки, які краще спонукають всіх брати на себе відповідальність, активізують і залучають творчий потенціал кожного і відкладають прийняття рішень. до останнього відповідального моменту.

FDD більш структурована, ніж інші методології. Це відображено у методах та прийомах, які використовуються.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

1. Чому (і як) ви повинні використовувати розробку на основі функцій [Електронний ресурс] – Режим доступу до ресурсу: https://www.lucidchart.com/blog/why-use-feature-driven-development.
2. Gregory P., Kruchten P. Agile Processes in Software Engineering and Extreme Programming, 2021. 414 c.
3. Feature Driven Development (FDD) and Agile Modeling [Електронний ресурс] – Режим доступу до ресурсу: http://www.agilemodeling.com/essays/fdd.htm.
4. Brechner E. Agile Project Management with Kanban (Developer Best Practices), 2015. 160 c.
5. Лагунова І. А. Сутність та принципи концепції ризик-менеджменту. Актуальні проблеми держав- ного управління. 2018. № 1 (53). С. 44–52.
6. Лагунова І. А. Сутність та принципи концепції ризик-менеджменту. Актуальні проблеми держав- ного управління. 2018. № 1 (53). С. 44–52.