Лабораторна робота №6

Методи розробки програмного забезпечення

Мета: Ознайомитися з основними методами розробки програмного забезпечення.

Теоретичні відомості

Методами розробки програмного забезпечення називається набір дій, який використовується для структурування, планування та управління процесом розробки інформаційної системи. Ці дії включають в себе попередню підготовку звітів та штучних об’єктів, які будуть створені та доведені до практичної реалізації командою розробників з метою розробки або обслуговування кінцевого продукту.

Виділяють наступні методології розробки програмного забезпечення:

- структурне програмування;
- Capgemini SDM або SDM2 (System Development Methodology);
- структурний метод системного аналізу та проектування;
- методологія програмних систем;
- об'єктно-орієнтоване програмування;
- швидку розробку програмного забезпечення;
- метод розробки динамічних систем;
- Scrum;
- процес групової розробки;
- екстремальне програмування.

Окрім методологій розробки розрізняють також моделі розробки ПЗ. Серед них:

1. модель водоспаду;
2. модель прототипів;
3. інкрементна модель
4. спіральна модель;
5. модель швидкої розробки;

Розглянемо кожну з них окремо.

1. МОДЕЛЬ ВОДОСПАДУ

Дана модель є моделлю послідовної (поетапної) розробки, в процесі якої проект проходить фази аналізу вимог, проектування, реалізації, тестування, інтегрування та обслуговування. Перехід до наступної фази відбувається лише після завершення попередньої. Оскільки повернення до попередньої фази не передбачається, такий підхід нагадує водоспад (рис. 6.1).

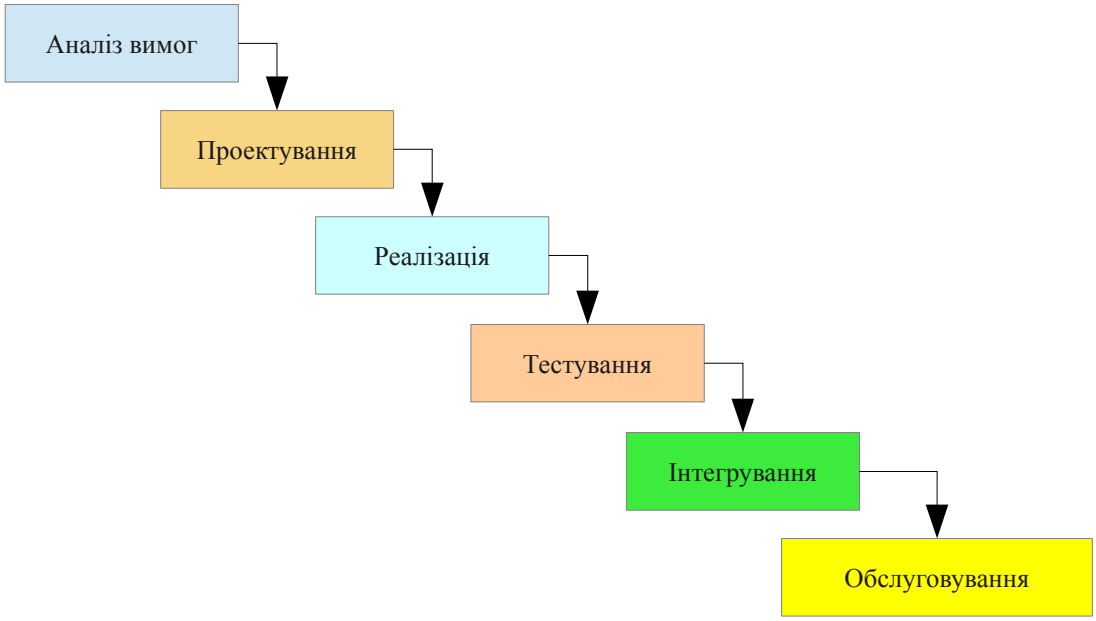

Рисунок 6.1. — Модель «водоспаду»

Вперше дану модель було запропоновано у 1970 році Вінстоном Ройсом як некоректну. Її недоліком є те, що на кожному з етапів багато даних можуть бути невідомими аж до завершення даного етапу. Якщо ж після завершення попереднього етапу з'являються нові дані, необхідно пройти всі фази з самого початку. Тому, ця модель може використовуватися лише для ПЗ з чітко поставленим технічним завданням, і вимоги до якого не будуть змінюватися в процесі розробки. Перевагою даної моделі є те, що вона робить наголос на чіткому документуванні кожного етапу, і, як наслідок, джерельного коду.

Базовими принципами даної моделі є:

1. поділ проекту на окремі фази, з можливістю незначного перекриття між фазами та поверненням до попереднього етапу;
2. наголос на плануванні, встановленні часових рамок, визначенні граничних термінів, бюджету та реалізації готового проекту. Всі ці елементи плануються одночасно.
3. чіткий контроль над проектом загалом завдяки детальній документації, яка генерується наприкінці кожної фази.

За кожним з етапів моделі закріплено набір певних дій:

1. *Аналіз* — збір даних про вимоги до системи;
2. *Проектування* — формалізація зібраних даних, узагальнена розробка майбутніх методів;
3. *Реалізація* — безпосереднє написання джерельного коду;
4. *Тестування* — перевірка кожного робочого блоку на предмет наявності помилок;
5. *Інтегрування* — об'єднання блоків у кінцеву програму;
6. *Обслуговування* — видача програми замовнику та подальший її супровід.

2. МОДЕЛЬ ПРОТОТИПІВ

Створення прототипів відрізняється від розробки готового продукту. Основна відмінність полягає у емуляції прототипом окремих функцій, в той час, як кінцева програма є завершеним продуктом. Внаслідок цього вже на перших етапах розробки можна отримати уявлення про те, як буде працювати кінцевий продукт, та внести необхідні зміни на ранніх етапах розробки ПЗ.

Розрізняють вертикальне масштабування прототипів та горизонтальне. При горизонтальному, розглядається система або підсистема загалом, із наголосом на взаємодію з користувачем. а не на реалізацію внутрішніх функцій. Цей тип масштабування використовується для представлення загального вигляду кінцевого продукту, проведення оцінки затрат на розробку, та можливості укладання угоди на випуск кінцевого продукту. При вертикальному масштабуванні більшу увагу приділяють окремо взятій підсистемі чи функції, із наголосом на реалізацію обумовлених методів.

Серед варіантів прототипів виділяють:

1. одноразові прототипи;
2. еволюційні прототипи;
3. інкрементні прототипи;
4. екстремальні прототипи.

Одноразові прототипи — використовуються лише для представлення кінцевого вигляду реалізованих блоків майбутнього ПЗ, а тому не приймають участі в подальшій розробці. Їх перевагою є швидке проектування, а основна задача, яка вирішуєтьс4я за допомогою таких прототипів — формування вимог до кінцевого проекту.

Еволюційний прототип базується на створенні чіткої структури, яка постійно покращується. Такі прототипи є функціональними блоками, які поступово набирають рис кінцевої системи. Перевагою даного типу прототипів є можливість програміста зосередитися на конкретному блоці, а не на системі загалом.

Інкрементний прототип переростає у кінцевий продукт, коли усі створені прототипи об’єднуються у одному проекті.

Екстремальні прототипи зазвичай використовуються при розробці веб-додатків. Вони складаються з трьох фаз: створення статичних веб-сторінок, реалізації функціональних можливостей емульованого рівня послуг; реалізації рівня послуг.

Серед переваг використання прототипів слід відзначити зменшення затрат на розробку кінцевого продукту та активну участь замовника на етапах розробки. Однак, ця модель має і недоліки, які проявляються у недостатньому аналізі усіх потреб замовника, складності переходу від одноразового прототипу до кінцевого продукту, потребі виділення додаткового часу на створення прототипу, а також відмінності у сприйнятті властивостей прототипу

розробником та замовником з відповідним проектуванням цих властивостей на кінцевий продукт.

3. ІНКРЕМЕНТНА МОДЕЛЬ

Модель інкрементної розробки зазвичай використовується разом із методами ітераційного проектування або ітераційної розробки. Така комбінація дозволяє мінімізувати ризики на кожному етапі розробки, а кількість циклів (рис. 6.2) визначається для кожного проекту окремо.

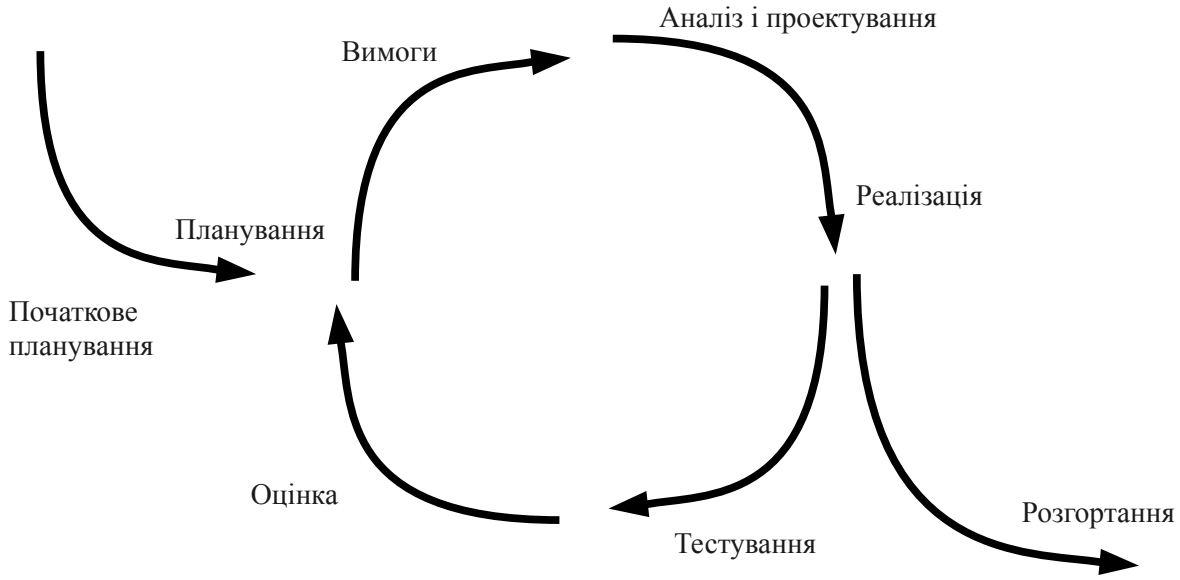

Рисунок 6.2 — Модель ітераційної розробки

Ітераційний метод базується на таких принципах як:

- використання моделі водоспаду для кожного невеликого блоку загальної системи, до того, як відбудеться перехід до наступної ітерації інкрементного методу;
- загальні вимоги визначаються до початку розробки, а надалі вони лише еволюціонують, використовуючи на кожній ітерації модель водоспаду;
- початкова концепція розробки ПЗ визначається за допомогою моделі водоспаду або ітераційного використання прототипів, які згодом формують кінцевий прототип робочої системи.

В свою чергу, інкрементна модель передбачає додавання невеликої частини функціональних блоків на кожній ітерації, після того, як попередній блок пройде повний цикл ітераційної розробки.

Ця модель є частиною таких методик як модифікована модель водоспаду, опис раціонально уніфікованого процесу та екстремального програмування, а також як обов’язковий компонент каркасів швидкої розробки ПЗ (agile software development).

4. СПІРАЛЬНА МОДЕЛЬ

Дана модель поєднує в собі властивості моделі «водоспаду» та моделі прототипів. Її основним призначенням є складні проекти з великим кошторисом.

Використання цієї моделі забезпечує випуск інкрементних версій продукту або інкрементне покращення поточних версій через певні проміжки часу. При цьому реалізація таких випусків відбувається по спіралі. Особливістю даної моделі є те, що вона включає в себе і управління ризиками при розробці ПЗ. Виявлення основних ризиків (як з точки зору програмування так і з точки зору управління) дозволяє зменшити їх вплив, та утримувати контроль над розробкою ПЗ.

Спіральна модель базується на постійному вдосконаленні продукту, з точки зору: визначення та аналізу вимог, системного та програмного проектування; а також, реалізації програмного коду. На кожній ітерації кола, поточний продукт є покращенням попередньої версії продукту даного етапу. Продукт проходить ті ж фази, що й у випадку моделі водоспадів, до яких додаються етапи планування, оцінки ризиків, створення прототипів та емуляція.

Додавання нових елементів відбувається тоді, коли вони стають готовими або ж про них стає відомо. Тому, спіральна модель не містить конфліктів, які можуть виникнути з попередніми вимогами та проектом. Даний метод базується на підходах, які використовують багато збирань (build) та випусків (release), а тому є придатним для переходу від циклу розробки до циклу обслуговування. Додатковою властивістю цієї моделі є залучення користувача на ранніх етапах розробки. Таке залучення є корисним коли мова йде про розробку елементів користувацької взаємодії, напр. зовнішній вигляд програми.

Починаючи від центру (рис. 6.3) кожен регіон спіралі проходить через декілька завдань:

- визначення цілей, альтернатив та обмежень для даної ітерації;
- оцінку альтернатив а також ідентифікацію та вирішення ризиків;
- розробку та верифікацію продукту даної ітерації;
- планування наступної ітерації.

При цьому визначення вимог відбувається на декількох етапах кількох ітерацій, так само, як і планування та оцінка ризиків. Кінцевий вибір проекту, реалізація, інтеграція та тестування відбувається на четвертій ітерації. Спіраль може повторюватися різну кількість разів для різних збирань. Завдяки цьому методу, деякі властивості продукту можна надати користувачу швидше, аніж у випадку використання моделі «водоспаду». Управління ризиками та невизначеністю даних є зручнішим у даному випадку, оскільки вводиться декілька точок прийняття рішень, а також допускається, що не всі вхідні дані можуть бути доступними на початковому етапі, і виникають лише при поділі задач на підзадачі.

Рисунок 6.3 — Спіральна модель розробки ПЗ

5. МОДЕЛЬ ШВИДКОЇ РОЗРОБКИ

Швидка розробка є моделлю, яка поєднує ітераційну розробку та створення прототипів. Даний термін описує принципи розробки закладений Джеймсом Мартіном у 1991. Його основними принципами є:

- основною метою є швидка розробка та передача високо-якісної системи при відносно низьких інвестиційних затратах;
- спроба зменшити наявні ризики, шляхом розбиття проекту на дрібніші сегменти та забезпечення більшої простоти при внесенні змін в процес розробки;
- зосередженість на швидкому створенні високоякісних систем, з використанням, в першу чергу, за допомогою ітеративної моделі прототипів (яка застосовується на будь-якій стадії розвитку), активного залучення користувачів і комп'ютерних засобів розробки. Ці засоби можуть включати в себе:

- ПЗ для проектування графічних інтерфейсів (GUI-builder);
- Computer-Aided Software Engineering (CASE) засоби;
- системи управління базами даних (СУБД);
- мови програмування четвертого покоління;
- генератори коду;
- об'єктно-орієнтовані методики.

- основний наголос робиться на задоволенні потреб бізнесу, а технологічні та інженерні знання мають менший пріоритет.
- управління проектом передбачає виставлення пріоритетів на розробку ПЗ та визначення термінів реалізації або «часових рамок». Якщо проект починає порушувати рамки, тоді акцент робиться на зниженні вимог до розміру часових рамок, а не на збільшенні загального терміну розробки.
- в загальних рисах, тут присутнє і спільне проектування ПЗ (joint application design, JAD), під час якого користувачі активно приймають участь у проектуванні системи, шляхом консенсусної розробки, яка визначається під час семінарів, або інших видів електронної взаємодії;.
- обов'язковою умовою є активна участь користувачів;
- здійснення ітеративної розробки промислового ПЗ, на противагу до одноразових прототипів.
- створення документації, яка необхідна для сприяння подальшого розвитку і обслуговування.
- проведення аналізу типових систем і методів проектування також може підпадати під дану модель.

Модель швидкої розробки складається з чотирьох етапів (рис. 6.4):

1. Етап планування вимог — об’єднує елементи системного планування та фази системного аналізу життєвого циклу розробки ПЗ. Таке планування здійснюється спільно користувачами, менеджерами та розробниками.
2. Користувацький етап проектування — на даному етапі користувачі взаємодіють з системними аналітиками, створюючи моделі та прототипи для всіх системних процесів, а також механізмів вводу/виводу.
3. Етап збирання — передбачає розробку програмного забезпечення. на даному етапі відбувається написання програмного коду, тестування його та інтеграція з готовими модулями.
4. Етап переносу — даний етап схожий на етап впровадження життєвого циклу розробки ПЗ. Тут відбувається перетворення даних, тестування, перехід на нову систему з відповідним навчанням користувачів.

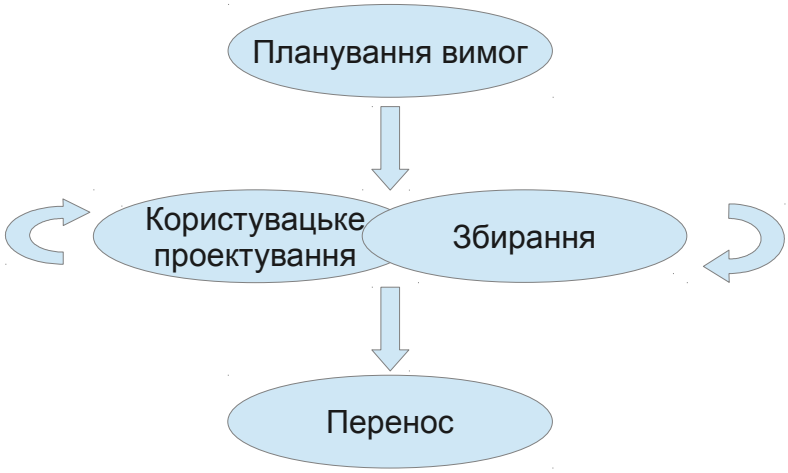

Рисунок 6.4. — Модель швидкої розробки

Хід роботи

1. Ознайомтеся із теоретичними відомостями.
2. Відповідно до свого варіанту проаналізуйте файл змін (Changelog) відповідного проекту програмного забезпечення.
3. На основі аналізу файлу змін, опишіть процес розробки, використовуючи ту модель розробки, яка, на вашу думку, найкраще підходить до розробки даного проекту.
4. Оформіть звіт по роботі. Звіт повинен містити короткий опис вказаного проекту, основні задачі, які він вирішує, обрану мову програмування, а також опишіть процес розробки даного проекту, використовуючи обрану Вами модель та файл змін проекту.

Контрольні запитання

1. Що таке метод розробки програмного забезпечення?
2. Які ви знаєте методи розробки ПЗ?
3. Перелічіть всі етапи розробки, які передбачено у моделі водоспаду.
4. В чому особливість моделі прототипів?
5. Що таке ітераційна розробка?
6. Чим відрізняється інкрементна модель від спіральної?
7. Які особливості застосування моделі швидкої розробки?

Варіанти завдань

1. http://sourceforge.net/projects/peazip/
2. http://sourceforge.net/projects/staruml/

3. http://sourceforge.net/projects/akelpad/
4. http://sourceforge.net/projects/free-cad/
5. http://sourceforge.net/projects/lazarus/
6. http://sourceforge.net/projects/nsis/
7. http://sourceforge.net/projects/mplayerosx/
8. http://sourceforge.net/projects/drjava/
9. http://sourceforge.net/projects/opencvlibrary/
10. http://sourceforge.net/projects/lame/
11. http://sourceforge.net/projects/audacity/
12. http://sourceforge.net/projects/synfig/
13. http://sourceforge.net/projects/goldendict/
14. http://sourceforge.net/projects/vlc/
15. http://sourceforge.net/projects/openofficeorg.mirror/
16. http://sourceforge.net/projects/freemind/
17. http://sourceforge.net/projects/scribus/
18. http://sourceforge.net/projects/adium/
19. http://sourceforge.net/projects/nagios/
20. http://sourceforge.net/projects/gnucash/
21. http://sourceforge.net/projects/sugarcrm/
22. http://sourceforge.net/projects/zencart/
23. http://sourceforge.net/projects/xdxf/
24. http://sourceforge.net/projects/openbravo/
25. http://sourceforge.net/projects/filezilla/
26. http://sourceforge.net/projects/azureus/
27. http://sourceforge.net/projects/pidgin/
28. http://sourceforge.net/projects/amule/
29. http://sourceforge.net/projects/clamav/
30. http://sourceforge.net/projects/psi/
31. http://sourceforge.net/projects/davmail/
32. http://sourceforge.net/projects/mysql-python/
33. http://sourceforge.net/projects/phpmyadmin/
34. http://sourceforge.net/projects/eclipse-cs/
35. http://sourceforge.net/projects/codeblocks.berlios/
36. http://sourceforge.net/projects/libsdl-android/
37. http://sourceforge.net/projects/jboss/
38. http://sourceforge.net/projects/ireport/
39. http://sourceforge.net/projects/wxpython/