**Лабораторна робота №3.**

**Тема:** Проектування програмного забезпечення за допомогою каскадної (водоспадної) моделі

**Мета:** Навчитись проектувати гіпотетичне програмне забезпечення, спираючись на принципи, що закладені в каскадній моделі.

**КОНТРОЛЬНІ ПИТАННЯ:**

1. Що таке модель конструювання?
2. Що таке життєвий цикл програмного забезпечення.
3. Які різновиди моделей життєвого циклу програмного забезпечення Вам відомі?
4. Назвіть основні характеристики і фази каскадної моделі.
5. Назвіть основні переваги каскадної моделі?
6. Назвіть основні недоліки каскадної моделі?

**Завдання:** Ознайомтесь з основними принципами каскадної моделі, за темою та сформуйте детальний перелік робіт щодо розробки програмного забезпечення згідно етапів каскадної моделі для власного проекту. Кожний етап повинен бути супроводжений документацією (визначити перелік документів, необхідний для кожного етапу).

**Методичні вказівки до виконання роботи:**

1. Робота повинна бути виконана згідно критеріїв оформлення документації на аркушах формату А4.
2. Студентам перед виконання лабораторної роботи потрібно ознайомитись із необхідним теоретичним матеріалом, потім обрати тему. В якості теми можна взяти тему попередніх лабораторних робіт (конструювання програмного забезпечення для оброблення результатів анкетування) або обрати власну тему, узгодивши її з викладачем.
3. Спочатку рекомендується проговорити можливі варіанти виконання.
4. Якщо потрібні пояснення – можна звернутись до викладача. Викладач сам вирішує відповідати на питання чи надати підказки, де можна знайти відповідь.
5. Протягом виконання лабораторної роботи студенти можуть використовувати конспекти або інші джерела інформації.
6. Роботу потрібно набирати на комп’ютері, що розташований в аудиторії або на власному ноутбукові, назвавши документ «Лабораторна робота № ».
7. По закінченню лабораторну роботу потрібно здати на перевірку викладачеві, роздрукувавши її або надіславши електронною поштою. Якщо викладач знаходить помилки чи неточності, він може повернути роботу на доопрацювання.

Оформлену відповідно до встановлених вимог роботу здати на перевірку викладачеві, надіславши електронною поштою на адресу [t.i.lumpova@gmail.com](mailto:t.i.lumpova@gmail.com). Якщо викладач знаходить помилки чи неточності, він може повернути роботу на доопрацювання.

Файл з роботою повинен мати назву в такому форматі:

**KPZ <Номер групи><Номер лекції / лабораторної> [літера позначення типу роботи L – лекція, R – лабораторна]<Прізвище англійською>**.. Наприклад, **KPZ4101R**buts.doc.

Тему в заголовку листа записати

**KPZ<Номер групи>-ЛР<Номер лабораторної>-<Прізвище >**

**Строк виконання цієї роботи ІПЗ-41 – 08.10.2021**

**ІПЗ-42 – 09.10.2021**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, Тему в заголовку листа записати

**KPZ <Номер групи>-Запитання-<Прізвище >**.

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

Класична каскадна модель, незважаючи на отриману останнім часом негативну оцінку, справно служила фахівцям з програмного інжинірингу багато років. Розуміння її сильних сторін і недоліків покращує оцінний аналіз інших, найчастіше більш ефективних моделей життєвого циклу, заснованих на даній моделі.

У перші роки практики програмування спочатку записувався програмний код, а потім відбувалася його налагодження. Загальноприйнятим вважалося правило починати роботу не з розробки плану, а з загального ознайомлення з продуктом. Без зайвих формальностей можна було спроектувати, закодувати, налагодити і протестувати ПО ще до того, як воно буде готове до випуску. Це нагадувало процес, зображений на рис. 1. У структурі такого процесу є декілька недоліків.

По-перше, оскільки спочатку не існувало офіційного проекту або аналізу, неможливо було впізнати про момент завершення процесу. Також був відсутній спосіб визначення відповідності вимогам щодо досягнення якості.

Выходы

Входы

Кодирование и тестирование

Делать, пока не будет сделано

***Рис. 1. Модель процесса "делать, пока, не будет сделано”***

У 1970 році каскадна модель була вперше визначена як альтернативний варіант методу розробки ПЗ за принципом кодування-усунення помилок, який був широко поширений в той час. Це була перша модель, яка формалізувала структуру етапів розробки ПЗ, надаючи особливого значення вихідним вимогам і проектування, а також створенню документації на ранніх етапах процесу розробки.

Початковий етап виконання каскадної моделі показаний у лівій верхній частині рис. 2. Продовження процесу виконання реалізується за допомогою впорядкованої послідовності кроків. У моделі передбачено, що кожна наступна фаза починається лише тоді, коли повністю завершено виконання попередньої фази Кожна фаза має певні критерії входу і виходу: вхідні і вихідні дані.

В результаті виконання генеруються внутрішні чи зовнішні дані проекту, включай документацію та ПЗ. Документи по аналізу вимог згодом передаються системним фахівцям, які в свою чергу передають їх розробника програмних систем більш високого рівня. Програмісти передають детальні технічні характеристики програмістам, які вже представляють готовий код тестерам.

Перехід від однієї фази до іншої здійснюється за допомогою формального огляду. Таким чином, клієнт отримує загальне уявлення про процес розробки, крім того відбувається перевірка якості програмного продукту. Як правило, проходження стадії огляду вказує на домовленість між командою розробників і клієнтом про те, що поточна фаза завершена і можна перейти до виконання наступної фази. Закінчення фази зручно приймати за стадію в процесі виконання проекту.

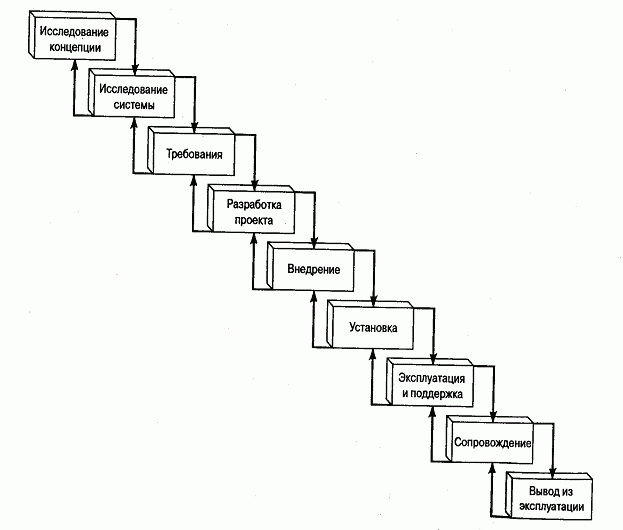


Рис.2. Каскадна модель проектування програмного забезпечення

В результаті завершення певних фаз формується базова лінія, яка в даній точці "заморожує" продукти розробки. Якщо виникає потреба в їх зміні, тоді для внесення змін використовується формальний процес змін.

У критичних точках каскадної моделі формуються базові лінії, остання з яких є базовою лінією продукту. Після формування заключній базової лінії проводиться огляд приймання.

***Короткий опис фаз каскадної моделі***

Наведена нижче характеристика являє собою короткий опис кожної фази каскадної моделі (включаючи фази інтеграції):

* *дослідження концепції* - відбувається дослідження вимог на системному рівні з метою визначення можливості реалізації концепції;
* *процес системного розподілу* - може бути пропущений для систем з розробки виключно ПЗ. Для систем, в яких необхідна розробка як апаратного, так і програмного забезпечення, необхідні функції застосовуються до ПЗ та обладнанню відповідно із загальною архітектурою системи;
* *процес визначення вимог* - визначаються програмні вимоги для інформаційної предметної області системи, призначення, лінії поведінки, продуктивність і інтерфейси. (В разі необхідності в процес також включено функціональний розподіл системних вимог до апаратному і програмному забезпеченню.);
* *процес розробки проекту* - розробляється і формулюється логічно послідовна технічна характеристика програмної системи, включаючи структури даних, архітектуру ПЗ, інтерфейсні уявлення і процесуальну (алгоритмічну) деталізацію;
* *процес реалізації* - в результаті його виконання ескізне опис ПЗ перетворюється на повноцінний програмний продукт. При цьому створюється вихідний код, база даних і документація, які лежать в основі фізичного перетворення проекту. Якщо програмний продукт являє собою придбаний пакет прикладних програм, основними діями по його реалізації будуть установка і тестування пакета програм. Якщо програмний продукт розробляється на замовлення, основними діями є програмування та код-тестування;
* *процес установки* - включає установку ПЗ, його перевірку і офіційну приймання замовником для операційного середовища;
* *процес експлуатації та підтримки* - має на увазі запуск користувачем системи і поточне забезпечення, включаючи надання технічної допомоги, обговорення питань, що виникли з користувачем, реєстрацію запитів користувача на модернізацію та внесення змін, а також коригування або усунення помилок;
* *процес супроводження* - пов'язаний з дозволом програмних помилок, несправностей, збоїв, модернізацією та внесенням змін, що генеруються процесом підтримки. Складається з ітерацій розробки і передбачає зворотний зв'язок з надання інформації про аномалії;
* *процес виведення з експлуата*ції - висновок існуючої системи з її активного використання або шляхом припинення її роботи, або завдяки її заміні новій системою або модернізованою версією існуючої системи;
* *інтегральні завдання* - включають початок роботи над проектом, моніторинг проекту та його управління, управління якістю, верифікацію та атестацію, менеджмент конфігурації, розробку документації та професійну підготовку протягом усього життєвого циклу.