**Лекція 3. Планування конструювання програмного забезпечення**

**Постановка завдання.** Під час перших зустрічей щодо замовлення на розроблення ПЗ замовник дуже приблизно уявляє, що йому потрібно. Без чіткого визначення процесів, для автоматизації яких буде використовуватись ПЗ, неможливо навіть приблизно оцінити обсяг робіт.

**Потреби** (needs) – відображають проблеми бізнесу, персоналій або процесу, що повинні співвідноситися з використанням або придбанням системи

Щоб тримати уявлення про можливі обсяги робіт, потрібно пропонувати замовнику надати або розробити технічне завдання. Завдяки цьому системні аналітики зможуть розібратися в задачі, виконати декомпозицію системи на компоненти, приблизно визначити обсяги цих компонентів і відповідно час їх реалізації.

Ця початкова стадія ЖЦ є "оцінкою здійсненності".

Постановка завдання – найбільш творча частина ЖЦ. Потрібно описати поведінку розроблюваної системи. Ця система отримує якісь сигнали з її оточення, тому треба описати поведінку оточення, але оточення само залежить і змінюється під впливом системи, її сигналів, особливо аварійних. Вирішують це протиріччя ітераційно, поетапно уточнюючи поведінку як системи, так і її оточення. Для відповідальних систем замовник може запропонувати розробити імітаційну модель системи та оточення.

У поставленому завданні замовник визначає вимоги до створюваної системи, які повинні задовольняти потреби користувачів і бути зрозумілими для розробників.

**Вимога** (Requirement) – умова або можливість, що визначена користувачем для вирішення проблеми або досягнення мети, та якій повинна відповідати або якою повинна володіти система чи її компонент, щоб задовольняти умови контракту, стандарту, специфікації або іншого формально репрезентованого документа. Вимоги поділяються на:

* вимоги користувача (User Requirements) – описують цілі/задачі користувачів системи, які повинні досягатися/виконуватися користувачами за допомогою створюваної програмної системи;
* функціональні вимоги (Functional Requirements) – вимоги, що конкретизують функції, які система або її компонент повинен виконувати;
* програмні вимоги (Software Requirements) – вимоги до створюваної системи, зрозумілі користувачами (замовниками) і розробниками (виконавцями) стосовно того, що робитиме система і чого від неї не варто чекати.

Оцінка складності системи, є сумою оцінок її компонентів, отриманих у результаті декомпозиції, значно точніша , ніж первісна оцінка системи в цілому.

Використання засобів формалізації результатів аналізу для їх документального оформлення також підвищує якість початкового опису вимог до системи.

**Формалізація постановки завдання** – розроблення послідовності моделей, кожна з яких описує систему та її оточення з різних точок зору і з поступовою деталізацією. Усі уявлення про систему, отримані в різних моделях, повинні збиратися в єдиному репозитарії (деякій спеціалізованій базі даних). Це полегшить наскрізне проектування, при якому кожна наступна модель використовує результати попередньої і ніяк їм не суперечить. Відповідно всі можливі перевірки повинні бути наскрізними.

Для якісного визначення вимог до ПЗ потрібно спочатку провести аналіз та сформувати їх специфікації.

**Аналіз вимог** (Requirements Analysis) – трансформація інформації, отриманої від користувачів (та інших зацікавлених осіб) у чітко та однозначно визначені програмні вимоги, що передаються інженерам для реалізації у програмному коді.

Аналіз вимог включає:

* виявлення і розв’язання конфліктів між вимогами;
* визначення меж задачі, що вирішується створюваним програмним забезпеченням;
* у загальному випадку – визначення меж (Scope) і змісту програмного проекту;
* деталізацію системних вимог для встановлення програмних вимог.

**Специфікація** (Specification) – документ, що в закінченій, точній і перевіреній формі описує вимоги, проект, поведінку або інші характеристики компонента або системи, а також процедури, спрямовані на визначення того, чи задовольняються описані характеристики. Для опису комплексних проектів (у частині вимог) використовують три основні специфікації:

* визначення системи (System Definition), або специфікація вимог користувачів (User Requirements Specification);
* системних вимог (System Requirements);
* програмних вимог (Software Requirements).

Специфікація програмних вимог (Software Requirements Specification – SRS) встановлює основні угоди між користувачами (замовниками) і розробниками (виконавцями) стосовно того, що робитиме система і чого від неї не варто чекати. Цей документ може включати процедури перевірки створеного ПЗ на відповідність вимогам, що висуваються (у т.ч. плани тестування), описи характеристик стосовно якості та методів його оцінювання, питань безпеки тощо.

Специфікація вимог користувачів (User Requirements Specification) або концепція (concept <of operation>) визначає високорівневі вимоги, часто – стратегічні цілі, для досягнення яких створюється програмна система. Важливо, що цей документ описує вимоги до системи з позицій предметної області – домену.

Специфікація системних вимог (System Requirements) – описує програмну систему в контексті системної інженерії. Зокрема високорівневі вимоги до програмного забезпечення, що містить кілька або багато взаємозв'язаних підсистем і застосувань. При цьому система може бути як цілком програмною, так і містити програмні та апаратні компоненти. У загальному випадку до складу системи може входити персонал, що виконує певні функції системи, наприклад, авторизацію виконання певних операцій з використанням програмно-апаратних підсистем.

При постановці завдання потрібно, щоб програмні вимоги були зрозумілі, зв'язки між ними прозорі, а зміст специфікації не допускав різночитань та інтерпретацій, через які програмний продукт не буде відповідати потребам зацікавлених осіб. Тому потрібні інструменти управління вимогами.

**Управління вимогами** (Requirements Management) – діяльність, виконання якої забезпечує опис вимог, відстежування їх змін, перевірки на несуперечливість і на порушення наперед визначених правил.

Від вхідної інформації про майбутній програмний продукт залежить те, яку методологію буде обрано в проекті з розроблення ПЗ. Методологій багато: і дуже формалізованих, і тих, що дають творчу свободу програмістам. Вибір методології обумовлюється досвідом керівника групи розробників та умовами, які встановлюють замовники до документування етапів робіт.

Методології розроблення ПЗ класифікуються за кількістю виконавців та критичністю проекту. Чим більше виконавців та/або вища критичність, тим більш формальна та регульована методика потрібна.

Вибір методу (методології) конструювання є ключовим аспектом для планування конструкторської діяльності. Такий вибір значущий для всієї конструкторської діяльності, а також необхідних умов її здійснення, визначаючи порядок відповідних операцій та рівень виконання заданих умов перед тим як почнеться конструювання або складові його дії. Наприклад, модульне тестування в ряді методів є частиною робіт, після написання відповідного функціонального коду, в той час, як ряд гнучких (agile) практик, наприклад, XP (до речі, першими почали використовувати такі методи верифікації коду), вимагають написання Unit-тестів до того, як пишеться відповідний код, що вимагає тестування.

Використовуваний підхід до конструювання впливає на можливість зменшення (в ідеалі - мінімізації) складності, готовності до змін і конструюванні з можливістю перевірки.

Планування конструкторської діяльності визначає порядок, в якому створюються компоненти та інші активи цієї фази діяльності, проводяться роботи по забезпеченню якості одержуваного програмного забезпечення, розподіляються завдання і відповідні ресурси, в тому числі, визначаються призначення / відображення робіт конкретним інженерам-програмістам, членам проектної групи. Все це, звичайно, відбувається, дотримуючись правил, що визначаються використовуваним методом (методологією, практиками і т.п.).

На елементарному рівні система, як правило, складається з небагатьох типів підсистем, по-різному скомбінованих і організованих. Ієрархії такого типу отримали назву ***«*ціле*-*частина*».***

Поведінка системи в цілому зазвичай виявляється складніше поведінки окремих частин, причому завдяки більш сильним внутрішнім зв'язкам особливості системи в основному обумовлені відносинами між її частинами, а не частинами як такими.

У природі існує ще один вид ієрархії *-* ієрархія *«***просте*-*складне***»* або ієрархія розвитку (ускладнення) систем в процесі еволюції. У цій ієрархії будь-яка функціонуюча система є результатом розвитку більш простої системи. Саме даний вид ієрархії реалізується механізмом успадкування об'єктно-орієнтованого програмування.

Будучи в значній мірі відображенням природних і технічних систем, програмні системи зазвичай є ієрархічними, тобто мають описані вище властивості. На цих властивостях ієрархічних систем будується блочно*-*ієрархічний підхід до їх дослідження або створення*.* Цей підхід передбачає спочатку створювати частини таких об'єктів (блоки, модулі), а потім збирати з них сам об'єкт.

Процес розбиття складного об'єкта на порівняно незалежні частини отримав назву **декомпозиції**. При декомпозиції враховують, що зв'язки між окремими частинами повинні бути слабшим, ніж зв'язки елементів усередині частин. Крім того, щоб з отриманих частин можна було зібрати об'єкт, що розробляється, в процесі декомпозиції необхідно визначити всі види зв'язків частин між собою.

При створенні дуже складних об'єктів процес декомпозиції виконується багаторазово: кожен блок, у свою чергу, декомпозується на частини, поки не отримують блоки, які порівняно легко розробити. Даний метод розробки отримав назву **покрокової деталізації***.*

В процесі декомпозиції намагаються виділити аналогічні блоки, які можна було б розробляти на загальній основі. Таким чином забезпечують збільшення ступеня повторюваності кодів і, відповідно, зниження вартості розробки. Результат декомпозиції зазвичай представляють у вигляді схеми ієрархії, на нижньому рівні якої розміщають відносно прості блоки, а на верхньому - об'єкт, що підлягає розробці.



На кожному ієрархічному рівні опис блоків виконують з певним ступенем деталізації, абстрагуючись від несуттєвих деталей. Отже, для кожного рівня використовують свої форми документації і свої моделі, що відображають сутність процесів, які виконуються кожним блоком. Так, для об'єкта в цілому, як правило, вдається сформулювати лише найзагальніші вимоги, а блоки нижнього рівня повинні бути специфіковані так, щоб з них дійсно можна було зібрати працюючий об'єкт. Іншими словами, чим більше блок*,* тим більш абстрактним має бути його опис При дотриманні цього принципу розробник зберігає можливість осмислення проекту і, отже, може приймати найбільш правильні рішення на кожному етапі, що називають **локальною оптимізацією** (на відміну від глобальної оптимізації характеристик об'єктів, яка для дійсно складних об'єктів не завжди можлива).

Потрібно мати на увазі, що поняття складного об'єкта в міру вдосконалення технологій змінюється, і те, що було складним вчора, не обов'язково залишиться складним завтра.

Отже, в основі блочно*-*ієрархічного підходу лежать декомпозиція і ієрархічне впорядкування*.* Важливу роль відіграють також такі принципи:

* несуперечність – контроль узгодженості елементів між собою;
* повнота – контроль на присутність зайвих елементів;
* формалізація – строгість методичного підходу;
* повторюваність – необхідність виділення однакових блоків для здешевлення і прискорення розробки;
* локальна оптимізація – оптимізація в межах рівня ієрархії.

Сукупність мов моделей, постановок завдань, методів описів деякого ієрархічного рівня прийнято називати **рівнем проектування***.*

Кожен об'єкт у процесі проектування, як правило, доводиться розглядати з кількох сторін. Різні погляди на об'єкт проектування прийнято називати аспектами проектування*.*

Крім того, що використання блочно-ієрархічного підходу дає можливість створення складних систем, він також:

− спрощує перевірку працездатності як системи в цілому, так і окремих блоків;

− забезпечує можливість модернізації систем, наприклад, заміни ненадійних блоків із збереженням їх інтерфейсів.

Використання блочно-ієрархічного підходу стосовно програмним системам стало можливим тільки після конкретизації загальних положень підходу та внесення деяких змін у процес проектування. При цьому структурний підхід враховує тільки властивості ієрархії «ціле-частина», а об'єктний - використовує ще й властивості ієрархії «просте-складне».

Відомо, що технологічний цикл конструювання ПЗ містить три процеси:

* 1) аналіз;
* 2) синтез;
* 3) супроводження.

Протягом аналізу виявляються цілі створюваної системи, тобто вказуються функції та зміни стану системи залежно від оточення та керованих параметрів.

У процесі синтезу вказуються способи реалізації запропонованих на першому етапі функцій системи, тобто виконується програмна реалізація системи. Виокремлюють три етапи синтезу:

* 1) проектування ПЗ;
* 2) кодування ПЗ;
* 3) тестування ПЗ.

**Етап проектування доповнює вимоги до ПЗ, які подані моделями аналізу: *інформаційною*, *функціональною моделями* та *моделлю поведінки*. *Інформаційна модель* описує інформацію, яку, за задумом, має обробляти ПЗ. *Функціональна модель* визначає перелік функцій оброблення. *Модель поведінки* фіксує бажану динаміку системи (режими її роботи). На виході з етапу проектування виконується *розроблення даних, архітектури* та *процедурне розроблення ПЗ.***

* ***Розроблення даних*** – це результат перетворення інформаційної моделі аналізу в [структури даних](http://www.vaglivo.org/tema-lekciyi-sistema-upravlinnya-bazami-danih-microsoft-access.html) для реалізації програмної системи.
* ***Розроблення архітектури*** – виокремлення основних структурних компонентів та фіксація зв’язків між ними.
* ***Процедурне розроблення*** – опис послідовності дій у структурних компонентах, тобто визначення їх умісту.

**Далі створюють тести програмних модулів, виконують *тестування* та *перевірку системи програмування*. На проектування, кодування і тестування відводиться 75 % вартості конструювання системи програмування.**

* Рішення, які приймаються протягом проектування, роблять цей процес провідним етапом процесу синтезу. Важливість проектування визначається ще і якістю. ***Проектування*** – етап, на якому «вирощують» якість розроблення системи програмування, це єдиний шлях, який забезпечує правильну трансляцію вимог замовника в кінцевий програмний продукт.

**Розроблення програмних модулів**

* Методології розроблення ПЗ корисні для розроблення великих складних продуктів або розподілених інформаційних систем.

**Розробляючи відносно невеликі програми або реалізуючи конкретний програмний модуль, достатньо притримуватися такої послідовності кроків:**

Розробляючи відносно невеликі програми або реалізуючи конкретний програмний модуль, достатньо притримуватися такої послідовності кроків:

* 1. Постановка завдання.
* 2. Обґрунтований вибір засобів розроблення (програмування).
* 3. Вибір методу розв’язання завдання.
* 4. Розроблення алгоритму рішення завдання.
* 5. Кодування засобами обраної мови програмування.
* 6. Верифікація й перевірка коректності.
* 7. Тестування програми.
* 8. Налагодження програми у випадку виявлення помилок.
* 9. Розроблення документації.
* 10. Експериментальна експлуатація.
* 11. Промислова експлуатація.

**Методи керування і планування проектом**

Відповідно до світової статистики не всі реалізовані програмні проекти завершуються успішно, 33% з них є провальними з таких причин:

– вимоги замовника не виконуються,

– проект не вклався у вартість,

– проект не вклався в заданий термін,

– етапи робіт виявилися неузгодженими один з одним,

– менеджер не орієнтує розробників на застосування новітніх методів і засобів програмування, планування й дотримання стандартних угод на застосування моделі ЖЦ.

Основні положення керування проектом, завдання й методи якого відпрацьовувалися на технічних проектах (наприклад, перший проект розробки лайнера для перевезення пасажирів з Європи в Америку), призвели до того, що Генрі Гант уперше запропонував діаграмну схему обліку часу виконання проекту.

Сьогодні ці завдання сформульовано в такий спосіб

– планування проекту й складання графіків робіт виконання проекту,

– керування проектними роботами і командою виконавців,

– керування ризиками,

– оцінювання продукту й використовуваних процесів з метою вдосконалення тощо.

На практиці процес керування проектом містить у собі:

1) визначення обсягу робіт в рамках задач проекту з урахуванням наявних ресурсів та обмежень;

2) розроблення стратегії реалізації цілей проекту з урахуванням ризиків та сприятливих можливостей;

3) вибір та обґрунтування моделі ЖЦ, адекватної розміру, складності та цілям проекту;

4) визначення обсягів ресурсів та вартості вирішення задач проекту шляхом оцінювання існуючих варіантів досягнення цілей проекту та з огляду на існуючі ризики та умови;

5) розроблення схеми розподілу робіт та стратегію керування ними;

6) підбір інструментальних і людських ресурсів, необхідних для виконання проекту;

7) складання плану-графіка проекту, що ґрунтується на проведеному розподілі робіт, оцінках технічної та організаційної інфраструктури системи;

8) визначення певних осіб та груп для виконання проекту в цілому;

9) введення в дію планів проекту, надання гарантії виконавцям, що вони забезпечені правилами і нормами процесу ЖЦ та відстеження просування робіт відносно до цих планів і строків;

10) аналіз вимог і усунення відхилень від плану їх виконання та запобігання несподіваних проблем, виявлених у проекті.

**Мережні методи планування і керування**.

Це методи призначені для керування роботами проекту за планами, орієнтованими на зменшення строків і раціональне використання ресурсів на ньому. Вони базуються на відповідних моделях, які сприяють побудові раціональних і оптимальних за деякими критеріями плану реалізації проекту і забезпечують чітке виконання цього плану з елементами прогнозування і пошуку найкращого рішення до нього.

У загальному випадку мережна модель відображає часткову упорядкованість робіт і може містить у собі такі характеристики, як вартість, час, ресурси тощо, які відносять до окремих робіт і до проекту в цілому проекту. Мережу проекту можливо розглядати як орієнтований скінчений граф без контурів і відображає відношення передування між роботами, які можливо поставить у відповідні дузі або вершині графа.

Формою подання мережної моделі є графова, таблична, діаграмна тощо. Головній сутністю усіх моделей – состав робіт і порядок їх виконання у часі. У таких моделях роботи характеризують процеси їх виконання у часі. Їм відповідають вершини, а дугам відношенням між роботами. В якості вершин могуть бути також події, що позначають номерами або іменами. Цим подіям відповідає завершення однієї або декількох робіт, які попереджують (вхідні) до неї події і що створює умови для початку наступних (вихідних) робіт. Подія, що не має вхідних робіт, зветься початковою, а подія, що не має вихідних робіт – кінцевою або завершеною. Перша подія є ціллю проекту і відповідає досягненню окремих підцілей проекту.

Різновидом таких моделей є імовірнісна, в якої параметрам робіт відповідає випадкова величина (наприклад аварія, неотримання потрібних ресурсів, фінансовий кризи тощо).

Головне призначення мережних моделей – оцінка фактичного і майбутнього стану проекту і вироблення керуючих дій, а також оцінка ефективності від цих дій та вибору кращих з них.

Відомими методами керування проектами є метод критичного шляху СРМ (Critical Path Method) та PERT (Program Evaluation and Review Technique). Останній метод відомий у СРСР як ПЕРТ, що був створений у 1958р. групою керування спеціальними проектами.

**Планування** – це процес розподілу й призначення ресурсів (матеріальних і людських) з урахуванням вартості й часу виконання проекту. Неадекватне планування може спричинити зрив проекту або отримання в середовищі проекту неадекватних результатів.

Планування й перепланування – найємнісна в часі частина керування проектом, починаючи з ранніх процесів проекту. Сучасні методики пропонують засоби, надаючи в розпорядження менеджерів проектів інструменти й методи, які дозволяють планувати реальні роботи і досягати їхнього виконання.

**Види планів організації проекту.** Планування робіт проекту є першим кроком, на якому створюють плани-графіки, проводять їхній облік, контроль та регулювання. Результатом планування є різні види планів, які відповідають усім видам організаційної діяльності в процесі виконання проекту. Взагалі складаються такі плани:

– керування проектом за методами критичного шляху СРМ, PERT або іншими;

– план-графік робіт з проектування і строків їх виконання за відповідними методами керування і планування;

– розподіл робіт між розробниками відповідно плану;

– план досягнення якості, а також плани верифікації, валідації й тестування для отримання результатів проектування;

– поставки і регулювання технічних, CASE і людських ресурсів на проекті;

– супроводження, змінювання деяких вимог і усунення різних недоліків.

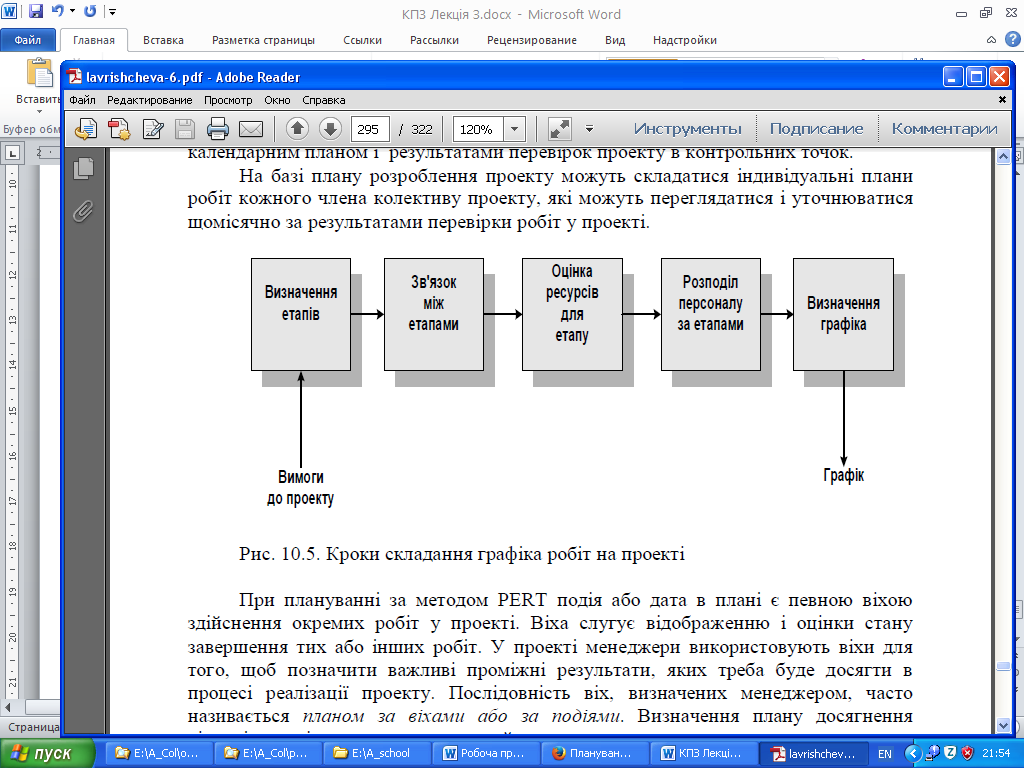
Керування проектом охоплює всі вказані плани і може мати в своєму складі додаткові плани, пов’язаніз деякими особливостями або вказівками замовника проекту.

При формуванні планів проекту встановлюють взаємозв’язок наведених планів з різних сторін розроблення та керування проектом. Процес планування починається на початку проекту, під час аналізу предметної області і визначення вимог до неї.

Тобто аналіз проекту виконується за методами СРМ, PERT, системного аналізу, аналізу витрат тощо. Цілі проекту розробляються таким чином, щоб оцінки продукту, були документованими задля подальшого їхнього моніторингу, види діяльності були затвердженими, а групи та розробники виконували підписані плани робіт з проекту.

Обґрунтування планів базується на реалістичних (якісних експертних) оцінках щодо робіт та їхнього затвердження керівником і замовником проекту. В плані проекту збалансовується сукупність інструментів, джерел даних, методологій, процесів і процедур, які забезпечують відповідні трудовитрати на запланованих роботах. У ньому враховуються зовнішнє середовище – інфраструктура організації, інструменти, людські ресурси, політика щодо до персоналу, ситуації на ринку тощо.

План проекту містить у собі календарний план, перелік документів та плану розроблення програмного продукту. Цей план враховує задану вартість, об’єм та план-графік робіт, відстеження ризиків і застосування затверджених методологій і інструментів розроблення проекту. Графік робіт складається за схемою, наведеною на рис. 10.5.



Після узгодження плану його з замовником їм користується менеджер.

Розроблений план проекту поновлюється протягом ЖЦ для проведення змін за календарним планом і результатами перевірок проекту в контрольних точок.

На базі плану розроблення проекту можуть складатися індивідуальні плани робіт кожного члена колективу проекту, які можуть переглядатися і уточнюватися щомісячно за результатами перевірки робіт у проекті.

При плануванні за методом PERT подія або дата в плані є певною віхою здійснення окремих робіт у проекті. Віха слугує відображенню і оцінки стану завершення тих або інших робіт. У проекті менеджери використовують віхи для того, щоб позначити важливі проміжні результати, яких треба буде досягти в процесі реалізації проекту. Послідовність віх, визначених менеджером, часто називається *планом за віхами або за подіями*. Визначення плану досягнення відповідних віх утворює календарний план.

Планування за методом СРМ або діаграми Ганта, які допомагають:

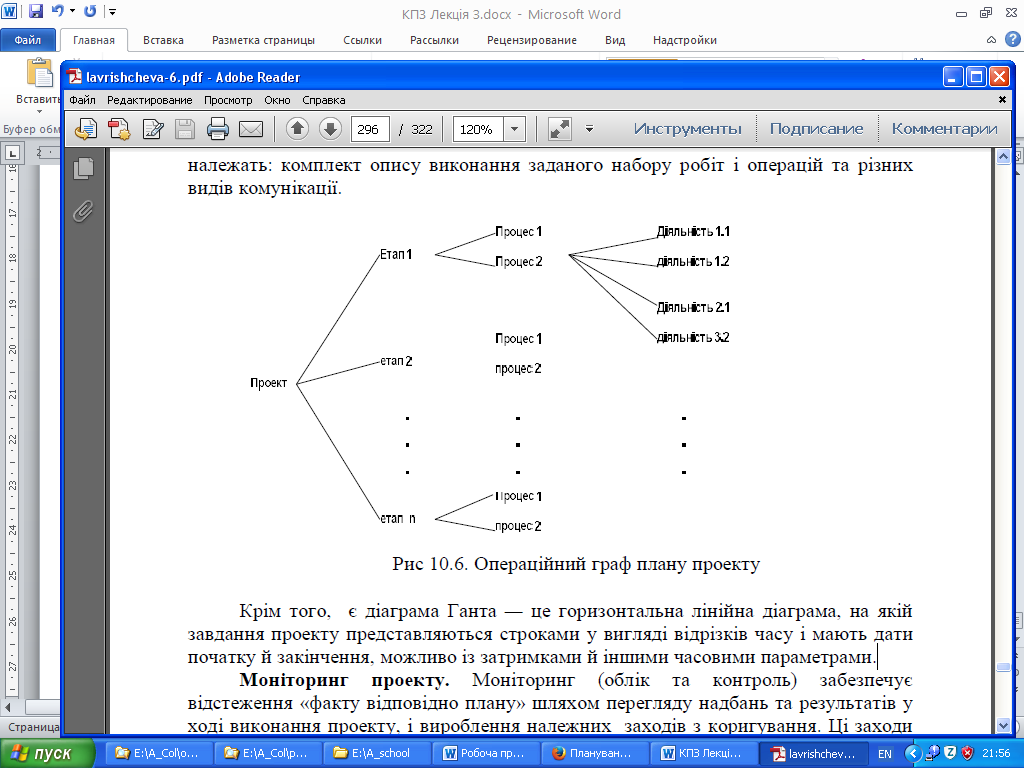
– структуризації робіт на основні компоненти й підкомпоненти;

– визначенню напрямів діяльності для досягнення комплексу цілей;

– розподілу відповідальних за виконання окремих робіт у проекті;

– отриманню звітності й узагальнення інформації про усякі дії проект.

План можна представляти етапами, станами і діяльностями (рис.10.6). У плані відображаються зв'язки між процесами, визначається інтервал часу для виконання кожної діяльності, час початку й завершення, а також опис різних видів демонстрацій функцій, надійності, захисту для замовника. До документів плану належать: комплект опису виконання заданого набору робіт і операцій та різних видів комунікації. Крім того, є діаграма Ганта *—* це горизонтальна лінійна діаграма, на якій завдання проекту представляються строками у вигляді відрізків часу і мають дати початку й закінчення, можливо із затримками й іншими часовими параметрами.



***Контрольні запитання для самоперевірки***.

1. Яка мета розроблення постановки завдання, в чому полягає її формалізація?
2. На які основні групи можна поділити вимоги?
3. Надайте визначення специфікації та охарактеризуйте її складові?
4. В чому полягає управління вимогами?
5. В чому полягає планування конструювання ПЗ?
6. Як застосовується блочно-ієрархічний підхід для конструювання ПЗ.
7. В чому сутність етапу проектування ПЗ?
8. Які етапи містить в собі етап керування проектами?
9. Назвіть і коротко охарактеризуйте відомі вам методи планування і керування проектами.

**Рекомендована література**

1. Лавріщева К.М. Програмна інженерія. Електронний підручник: http://csc.knu.ua/uk/library/books/lavrishcheva-6.pdf
2. Лавріщева К.М. Програмна інженерія. – Підручник.–К.:Академперіодика, 2008.–415с.
3. И. Соммервиль. Инженерия программного обеспечения,  
    6 изд. – И.д. "Вильямс", 2002.

***Для самостійного вивчення***: Ознайомитись з методами СРМ та PERT за [1, 2]. Поглибити матеріал лекції за наданою літературою. Вивчення лекційного матеріалу та додаткових джерел. Розгляд запитань і виконання завдань для самостійної роботи, запропонованих на лекції.