ЖДТУ	**Міністерство освіти і науки України** **Житомирський державний технологічний університет**

Затверджено науково-методичною
радою ЖДТУ
протокол № 5 від „30” січня 2018 р.

Методичні рекомендації
для самостійної роботи студентів
з навчальної дисципліни

«МОДЕЛЮВАННЯ ТА АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»

для студентів освітнього рівня «БАКАЛАВР»
денної форми навчання
напрям 6.050103 «Програмна інженерія»
освітньо-професійна програма «Програмна інженерія»
факультет інформаційно-комп’ютерних технологій
кафедра інженерії програмного забезпечення

Розглянуто і рекомендовано
на засіданні кафедри інженерії програмного забезпечення

Протокол № 4
від „23” квітня 2018 р.

Розробники: ст.викладач Власенко О.В., к.пед.н., доцент кафедри інженерії програмного забезпечення Ковальчук В.Н., к.т.н., доцент кафедри інженерії програмного забезпечення І.І. Сугоняк

Житомир, 2018

ЗМІСТ

ВСТУП

Сучасна індустрія розробки програмного забезпечення вимагає від майбутніх програмістів не лише знань з основних мов програмування та комп'ютерної техніки, але й розвинених аналітичних здібностей у процесі життєвого циклу проектування ПЗ. Складність сучасного програмного забезпечення призводить до збільшення ролі власне розробки та поступової деталізації моделі майбутнього програмного продукту, яка служить не лише засобом візуалізації абстрактних сутностей та їх поведінки, але засобом комунікації між розробником та замовником, між керівником проекту та командою програмістів.

Курс присвячений вивченню основ нотації уніфікованої мови моделювання або, скорочено, мови UML, який призначений для опису, візуалізації і документування об'єктно-орієнтованих систем та бізнес-процесів з орієнтацією на їх подальшу реалізацію у вигляді програмного забезпечення. Вивчення матеріалу курсу спрямоване на формування та вдосконалення знань з методології опису, візуалізації і документування об'єктно-орієнтованих систем за допомогою мови UML. Отримані в ході вивчення курсу знання можуть бути успішно використані в подальшому при управлінні проектами в ході розробки інформаційних моделей і програмних додатків.

Курс має своєю основною метою навчити студентів базових конструкцій мови UML використовувати різноманітні засоби роботи з використовувати CASE-засоби з метою автоматизації виконання всіх етапів концептуального, логічного та фізичного проектування архітектури програмних додатків.

При викладанні курс «Моделювання ПЗ» ґрунтується на знаннях студентів, які одержані під час вивчення дисциплін на попередніх курсах, таких як об'єктно-орієнтоване програмування, технології проектування програмного забезпечення.

1 МЕТОДОЛОГІЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОЕКТУВАННЯ ТА АНАЛІЗУ

Мета – повторити та узагальнити методологію об'єктно-орієнтованого проектування та аналізу.

Оскільки викладання даного курсу ґрунтується на знаннях студентів, які одержані під час вивчення дисциплін на попередніх курсах, таких як об'єктно-орієнтоване програмування, технології проектування програмного забезпечення, то завданням даного самостійного завдання є повторення та узагальнення попереднього вивченого матеріалу.

Після вивчення даної теми студенти повинні розуміти основні ознаки складних систем, причини складності програмних систем, математичні основи та етапи становлення об'єктно-орієнтованого підходу, історія розвитку мов програмування з точки зору ООП, особливості, принципи та переваги ОО аналізу та проектування, повторять основні поняття: клас, об'єкт, відношення між класами та об'єктами.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Контрольні запитання

Рекомендована література: [5, гл.1-2], [8, с. 5 - 6]

1. Що таке складність, що притаманна програмному забезпеченню? Опишіть основні причини складності ПЗ.
2. Порівняйте методи аналізу і декомпозиції систем: алгоритмічну (структурну) та об'єктно-орієнтовану. В чому переваги останньої?
3. В чому суть проектування складних систем. Для чого необхідно створення об'єктноорієнтованої моделі у процесі проектування.
4. Наведіть визначення об'єктно-орієнтованого програмування, проектування, аналізу. Що являється базисом ООП методології?
5. В чому переваги ООПА підходу над структурним підходом?
6. Опишіть розвиток мов програмування з точки зору розвитку ООП. Який вклад у розвиток ООП належить різним мовам програмування третього покоління?
7. Діаграми структурного програмування.

Теми рефератів.

1. "Срібної кулі немає". Обговорення стат'ї Фредеріка Брукса сорок років потому. [7, гл. 16-19]
2. Математичні основи об'єктно-орієнтованого аналізу та проектування. (Теорія множин. Теорія графів. Семантичні мережі). [1, с. 5 - 6].
3. Передісторія об'єктно-орієнтованого аналізу та проектування. (Діаграми функціонального моделювання. Діаграми сутність-зв'язок. Діаграми потоків даних). [1, с. 5 - 6].
4. Основні етапи розвитку UML. [2, лек. 1]

2 ОСОБЛИВОСТІ ПРОЦЕСУ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОЕКТУВАННЯ ТА АНАЛІЗУ

Мета – повторити та узагальнити знання щодо процесу об'єктно-орієнтованого проектування та аналізу.

Оскільки викладання даного курсу ґрунтується на знаннях студентів, які одержані під час вивчення дисциплін на попередніх курсах, таких як об'єктно-орієнтоване програмування, технології проектування програмного забезпечення, то завданням даного самостійного завдання є повторення та узагальнення попереднього вивченого матеріалу.

Після повторення даної теми студенти повинні пригадати особливості етапи об'єктноорієнтованого процесу розробки програмних систем, звернути увагу на роль моделювання та види моделей, які розробляються на кожному з етапів ОО проектування, повторити особливості метрик, управління ризиком і тестування за ООП методологією.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Контрольні запитання

Рекомендована література: [5, гл.3],[9, гл.15].

1. Які риси характеризують вдалі проекти? Які ознаки гарного архітектурного рішення проекту?
2. Що Ви розумієте під раціональним процесом проектування?
3. Що Ви розумієте під еволюційно-ітеративним циклом розвитку проекту?
4. Опишіть основні етапи процесу розробки (початок, розвиток, конструювання, перехід). Поясніть як співвідносяться потоки процесу розробки та ітерації.
5. Опишіть потоки процесу розробки (збір відомостей, аналіз, проектування, реалізація, тестування). Поясніть яку роль у процесі розробки грають моделі та моделювання.
6. Опишіть особливості кожного з технічних артефактів (набір вимог, реалізації, проектування, розміщення).
7. Як співвідносяться мікропроцес та мікропроцес об'єктно-орієнтованої розробки?
8. Опишіть етап мікропроцесу проектування, що пов'язаний з ідентифікацією об'єктів та даних на даному рівні абстракції. Які основні види діяльності на цьому етапі?
9. Опишіть етап мікропроцесу проектування, що пов'язаний з виявленням семантики класів і об'єктів. Які основні види діяльності на цьому етапі?
10. Опишіть етап мікропроцесу проектування, що пов'язаний з виявленням зв'язків між класами та об'єктами. Які основні види діяльності на цьому етапі?
11. Опишіть етап мікропроцесу проектування, що пов'язаний з реалізацією класів та об'єктів. Які основні види діяльності на цьому етапі?
12. Яка мета першого кроку мікропроцесу ООП – концептуалізації, які основні види діяльності на цьому кроці?

13. Яка мета другого кроку мікропроцесу ООП – аналізу, які основні види діяльності на цьому кроці?
14. Яка мета третього кроку мікропроцесу - проектування, які основні види діяльності на цьому кроці?
15. Яка мета четвертого кроку мікропроцесу проектування - еволюції, яка меті і основні види діяльності на цьому кроці?
16. Яка мета четвертого кроку мікропроцесу проектування - супроводу, яка меті і основні види діяльності на цьому кроці?

Теми рефератів.

1. Управління ризиком у процесі об’єктно-орієнтованого проектування.
2. Особливості метрик об’єктно-орієнтованих програмних систем.
3. Метрики об’єктно-орієнтованого тестування.
4. Приклад об’єктно-орієнтованого процесу розробки програмного забезпечення.
5. Особливості об’єктно-орієнтованого тестування.
6. Проектування об’єктно-орієнтованих тестових варіантів.
7. Моделі реалізації об’єктно-орієнтованих програмних систем.

3 КЛАСИ ТА ОБ'ЄКТИ. КЛАСИФІКАЦІЯ

Мета – повторити та узагальнити методологію об'єктно-орієнтованого проектування та аналізу.

Оскільки викладання даного курсу ґрунтується на знаннях студентів, які одержані під час вивчення дисциплін на попередніх курсах, таких як об'єктно-орієнтоване програмування, технології проектування програмного забезпечення, то завданням даного самостійного завдання є повторення та узагальнення попереднього вивченого матеріалу.

Після вивчення даної теми студенти повинні розуміти основні поняття: клас, об'єкт, відношення між класами та об'єктами, а також основні правила вибору відношень, операцій та реалізації. Студенти повинні також усвідомити види та роль класифікації у об'єктноорієнтованому аналізі та проектуванні.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Контрольні запитання

Рекомендована література: [8, с. 7- 18], [5, гл. 2-3], [9, гл. 9].

1. Опишіть основні принципи ОО підходу (абстрагування, модульність, ієрархія, обмеження доступу). Які переваги та особливості надає ОО підходу кожен з цих принципів?
2. Опишіть додаткові принципи ОО підходу (типізація, збереження, паралелізм). Які переваги та особливості надає ОО підходу кожен з цих принципів?
3. Природа класу та об'єкту. Спільне та відмінне.
4. Об'єкти: стан, поведінка, операції, ідентичність. Час життя об'єктів.
5. Відношення між об'єктами. Ролі об'єктів. Зв'язки, синхронізація, агрегація.
6. Природа класів. Життєвий цикл класів. Інтерфейс та реалізація.
7. Відношення між класами. Наслідування, асоціація, агрегація, використання, інстанціювання, метаклас.
8. Відношення між класами та об'єктами. Роль класів та об'єктів у аналізі та проектуванні.
9. Якість класів та об'єктів. Критерії: зчеплення, зв'язність, повнота, достатність, примітивність.
10. Правила вибору операцій. Функціональність. Пам'ять.
11. Правила для визначення типу відношень. Співробітництво. Механізми та видимість.
12. Вибір реалізації. Представлення. Модульна структура.
13. Завдання класифікації. Підходи до класифікації: розподіл по категоріям, концептуальна кластеризація, теорія прототипів. Роль класифікації в ООП.
14. Класичний підхід до ідентифікації класів та об'єктів.
15. Сучасні підходи до класифікації: аналіз поведінки, предметної області, варіантів, CRC картки.
16. Класифікація через ключові абстракції та механізми.

Теми рефератів.

1. Розвиток теорій класифікацій у філософії (від Платона і Арістотеля до Фоми Аквінського, Лока, Декарта).
2. Методологія системного аналізу і системного моделювання та її використання у процесі розробки програмного забезпечення.
3. Принципи об’єктно-орієнтованих представлення програмних систем.
4. Методи декомпозиції моделі на модулі та основні властивості модулів програмної системи.
5. Класичні методи структурного проектування програмних систем.
6. Поглиблене вивчення класів у мові UML. *[6, гл.10]*
7. Поглиблене вивчення відношень у мові UML. *[6, гл.11]*

4. ПОНЯТТЯ ВІЗУАЛЬНОГО ПРОГРАМУВАННЯ

Мета – ознайомитися з поняттям візуального програмування.

Після вивчення даної теми студенти повинні розуміти основні поняття: візуальне програмування, CASE-засіб, множинність точок зору при візуальному моделюванні, семантичний розрив між моделлю і програмним кодом, “невидимість” ПО, графи та діаграми візуальних моделей.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Контрольні запитання

Рекомендована література: [4, лек. 1]

1. Яка відмінність між засобами проектування у інженерії загалом та у програмній інженерії. Роль креслень.
2. Визначення візуального моделювання. Засоби візуального моделювання.
3. Програмні засоби візуального моделювання: універсальні та предметно-орієнтовані CASE-засоби.
4. Місце мов візуального моделювання у еволюції засобів програмування.
5. У чому полягає семантичний розрив візуальних моделей і програмного коду і його наслідки.
6. Поясніть терміни предметна область, модель, метамодель і метаметамодель?
7. Поясніть, що являють собою чотири рівня візуального моделювання. Чим вони відрізняються?
8. З чим пов'язано використання множинності точок зору при візуальному моделюванні ПЗ?
9. Навіщо для візуальних моделей виділяти граф моделі та діаграми? Що таке браузер моделі і навіщо він потрібен?
10. Розкажіть про операції над графом моделі. Розкажіть про операції над діаграмами. Розкажіть про поєднанні операцій над діаграмами з операціями над графом моделі.
11. Що таке репозиторій CASE-пакета? Розкажіть про способи його реалізації. Розкажіть про операції над графом моделі через браузер і засобами стороннього додатка (через відкритий програмний інтерфейс).

Теми рефератів.

1. Порівняльна характеристика найпоширеніших CASE-засобів.
2. Історія розвитку CASE-засобів і їх перспективи на майбутнє.
3. Історія розвитку мов візуального проектування та об’єктно-орієнтованого моделювання.

5. ЕЛЕМЕНТИ ГРАФІЧНОЇ НОТАЦІЇ ВАРІАНТІВ ВИКОРИСТАННЯ.

Мета – повторити та узагальнити теоретичний матеріал щодо діаграм прецендентів використання.

Після повторення даної теми студенти повинні знати, які основні графічні елементи використовується на діаграмах прецендентів, які типи зв'язків та стереотипів використовуються на цих діаграмах, хто може бути актором, на якому етапі проектування і з якою цілю використовуються ці діаграми.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Контрольні запитання

Рекомендована література: [2, лек. 3-4], [6, гл.17]

1. До якого типу моделей системи, що проектується відносяться діаграма варіантів використання?
2. Для яких цілей використовують діаграми варіантів використання?
3. Які особливості позначень відношення на діаграмі варіантів використання?
4. Які графічні примітиви використовуються на діаграмі прецедентів?
5. Як можна формалізувати функціональні вимоги до системи за допомогою діаграми варіантів використання?
6. Що так вимога, що таке сценарій використання. Наведіть приклади.
7. Які особливості специфікації функціональних вимог на діаграмі варіантів використання?
8. Наведіть приклад рекомендації з розробки діаграм варіантів використання.

Теми рефератів.

1. Особливості використання зв'язків та стереотипів на діаграмі Use Case.
2. Стереотипи у мові UML. Порівняння версій UML 1.0 та 2.0.

6. ОГЛЯД CASE-ЗАСОБІВ ДЛЯ ПОБУДОВИ UML ДІАГРАМ

Мета – розшири знання щодо класифікації та основних можливостей сучасних CASE засобів.

Після вивчення даної теми студенти повинні знати, які можливості є у сучасних CASE засобів щодо побудови діаграм UML, знати критерії вибору таких засобів, як вбудованих у середовища розробки так і незалежних, знати на яких етапах розробки програмного продукту доцільно CASE засоби і для чого, розуміти поняття кодогенерації, від чого залежить йогто якість та обмеження автоматичної кодогенерації.

Вивчення даної теми доцільно починати з огляду вказаної літератури, на основі чого скласти план конспекту чи реферату, при цьому слід тримати на увазі список питань на які необхідно звернути увагу при вивченні даної теми. Результатом даної самостійної роботи є конспект чи реферат, який оцінюється викладачем.

Порядок виконання роботи

1. Вивчення даного питання у вказаних джерелах.
2. Пошук відповіді на контрольні питання.
3. Складання плану конспекту чи реферату.
4. Написання конспекту чи реферату.

Орієнтовний план.

1. IBM Rational Rose
2. Borland Together
3. Microsoft Visio
4. Sparx Systems Enterprise Architect
5. Gentleware Poseidon
6. SmartDraw
7. Діаметр
8. Telelogic TAU G2
9. StarUML
10. Інші програми

Контрольні запитання

Рекомендована література: [3, лек.7], [8, с.44-55]

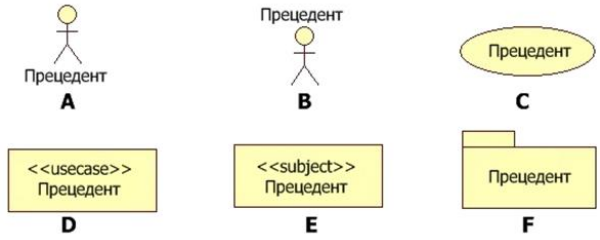
1. До якого типу моделей системи, що проектується відносяться діаграма варіантів використання?
2. Для яких цілей використовують діаграми варіантів використання?
3. Які особливості позначень відношення на діаграмі варіантів використання?
4. Які графічні примітиви використовуються на діаграмі прецедентів?
5. Як можна формалізувати функціональні вимоги до системи за допомогою діаграми варіантів використання?
6. Що так вимога, що таке сценарій використання. Наведіть приклади.
7. Які особливості специфікації функціональних вимог на діаграмі варіантів використання?
8. Наведіть приклад рекомендації з розробки діаграм варіантів використання.

Теми рефератів.

1. Можливості CASE засобу IBM Rational Rose по побудові UML діаграм.
2. Можливості CASE засобу Borland Together по побудові UML діаграм.
3. Можливості CASE засобу Microsoft Visio по побудові UML діаграм.
4. Можливості CASE засобу Umbrello Modeling UML по побудові UML діаграм.

7. ТЕСТИ ДЛЯ САМОПЕРЕВІРКИ ЗНАНЬ

1. Що таке метамодель мови UML?
А. Концептуальна модель
Б. Модель мови опису моделей
В. Модель реалізації
Г. Модель проектування
Д. Модель всесвіту
2. Яка характеристика тексту є значимою в UML-діаграмах?
А. Колір
Б. Накреслення
В. Розмір
Г. Міжсимвольний інтервал
Д. Регістр
3. Що таке кодогенерація?
А. Генерація текстової специфікації з існуючого коду
Б. Генерація коду із існуючої UML-моделі
В. Генерація тексту із існуючої UML-моделі
Г. Генерація UML-моделі з виконуваного файлу
Д. Генерація виконуваних файлів із існуючої UML-моделі
4. Окремим випадком якої діаграми є діаграми діяльності?
А. Діаграми прецедентів
Б. Діаграми кооперації
В. Діаграми послідовності
Г. Діаграми станів
Д. Діаграми розгортання
5. Що означає кратність 3..11, що вказана на одному з кінців асоціації?
А. Від 3 до 11 класів асоційовано з іншим класом
Б. 3 або 11 об’єктів одного класу асоційовано з одним об’єктом іншого класу
В. Від 3 до 11 включно об’єктів одного класу асоційовано з одним об’єктом іншого класу
Г. Менше 3 або більше 11 об’єктів одного класу асоційовано з одним об’єктом іншого класу
Д. 3 або 11 класів асоційовано з одним класом
6. Яким чином об’єкти всередині програми взаємодіють один з одним за об’єктного підходу?
А. Шляхом відправлення та отримання повідомлень
Б. Шляхом прямого виклику операцій один одного
В. Шляхом обміну інформації через буфер обміну
Г. Шляхом використання протоколу TCP/IP
Д. Шляхом прямого запису у пам’ять
7. Які види ліній використовується у зв’язках діаграм UML?
А. Пунктирна та суцільна
Б. Хвиляста та суцільна
В. Подвійна та пунктирна
Г. Штрихпунктирна та суцільна
Д. Суцільна та подвійна
8. Які з вказаних нотацій НЕ послужила основою для створення UML?
А. Booch
Б. Objectory
В. HTML
Г. OMT
Д. всі перераховані у А,Б,В,Г
9. В яких відношеннях НЕ можуть бути прецеденти?
А. Асоціація
Б. Генералізація
В. Композиція
Г. Включення
Д. Розширення
10. Виберіть вірне позначення прецедента



А. A
Б. B
В. C
Г. D
Д. E, F

11. Що таке композитний об’єкт на діаграмі кооперації?
А. Набір об’єктів одного класу
Б. Високорівневий об’єкт, що складається з декількох частин - об’єктів
В. Об’єкт, що володіє власним потоком керування і може ініціювати виконання дії іншими об’єктами
Г. Об’єкт, що володіє даними, але не може ініціювати виконання
Д. Екземпляр класу, що є агрегацією об’єктів інших класів

12. Що таке активний об’єкт на діаграмі кооперації?
А. Набір об’єктів одного класу
Б. Високорівневий об’єкт, що складається з декількох частин - об’єктів
В. Об’єкт, що володіє власним потоком керування і може ініціювати виконання дії іншими об’єктами
Г. Об’єкт, що володіє даними, але не може ініціювати виконання
Д. Екземпляр класу, що є агрегацією об’єктів інших класів

13. Що таке пасивний об’єкт на діаграмі кооперації?
А. Набір об’єктів одного класу
Б. Високорівневий об’єкт, що складається з декількох частин - об’єктів
В. Об’єкт, що володіє власним потоком керування і може ініціювати виконання дії іншими об’єктами
Г. Об’єкт, що володіє даними, але не може ініціювати виконання
Д. Екземпляр класу, що є агрегацією об’єктів інших класів

14. Як співвідносяться поняття моделі та діаграми?
А. Діаграми – графічне представлення сукупності елементів моделі
Б. Ці поняття є синонімічні
В. Поняття діаграма і модель не співвідносяться
Г. Будь-яка діаграма є основою для побудови моделі
Д. Ці поняття є антонімічні

15. Що означає стрілка, що зображена на одному з кінців ліній, що з’єднує актора і прецедент?
А. Вона направлена до того, чиїми послугами користуються
Б. Вона показує порядок виконання прецедентів
В. Вона вказує підпорядкований елемент
Г. Вона направлена до того, хто користуються послугами іншого
Д. Вона задає порядок читання діаграми

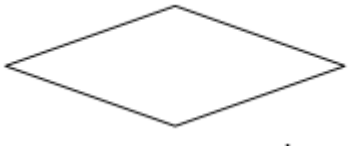
16. Яке з приведених визначень моделі є найбільш відповідним розумінню моделі в UML?
А. Модель – візуальне представлення фізичної системи в формі зображення
Б. Модель – абстракція фізичної системи, що розглядається з деякої точки зору і представленої на деякій мові чи графічній формі
В. Модель логічне уявлення фізичної системи в формі математичного рівняння чи коду
Г. Модель – деяке спрощене уявлення фізичної системи, яке дозволяє виділити основні суттєві її складові та зв’язки між ними
Д. Особа – що являє собою еталоного користувача системи

17. Яке з наведених нижче визначень принципу поліморфізму є вірним в контексті об’єктного підходу до проектування систем?
А. Поліморфізм характеризує властивість об’єктів приймати однакові форми в залежності від обставин
Б. Поліморфізм характеризує загальний принцип незалежності інтерфейсу операцій від особливостей їх реалізації у конкретному класі

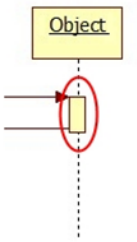
В. Поліморфізм характеризує закриття окремих деталей внутрішнього устрою класів від зовнішніх по відношенню до нього об’єктів чи користувачів
Г. Поліморфізм характеризує загальний принцип ООП у відповідності з яким знання про більш загальну категорію дозволяється застосовувати для більш часткової категорії
Д. Такого поняття не існує
18. Яке з визначень інкапсуляції є вірним в контексті об’єктного підходу до проектування систем?
А. Інкапсуляція характеризує властивість об’єктів приймати різні форми в залежності від обставин
Б. Інкапсуляція характеризує загальний принцип ООП у відповідності з яким знання про більш загальну категорію дозволяється застосовувати для більш часткової категорії
В. Інкапсуляція характеризує закриття окремих деталей внутрішнього устрою класів від зовнішніх по відношенню до нього об’єктів чи користувачів
Г. Інкапсуляція характеризує можливість ОПП визначати рівень доступу до окремих деталей внутрішнього устрою класів для інших зовнішніх чи наслідуваних класів
Д. Такого поняття не існує
19. Хто з вказаних людей не приймав безпосередню участь у розробці перших версій мови UML?
А. Граді Буч
Б. Джеймс Рембо
В. Джон фон Нейман
Г. Айвар Джекобсон
Д. всі перелічені у варіантах А,Б.В,Г
20. Що таке інтерфейс у UML?
А. Графічне представлення класу
Б. Логічна група елементів керування для роботи з об’єктом
В. Логічна група відкритих (public) операцій об’єкту
Г. Група операцій об’єкту, які надають доступ атрибутів класу
Д. Пристрій для інтеграції мікроконтролера до РС
21. Яким терміном UML можна описати людину, що здійснює покупку в онлайновому магазині?
А. Клієнт
Б. Зовнішня система
В. Суб’єкт
Г. Актор
Д. Замовник
22. Як на діаграмах прецедентів показується включення прецедентів? В вигляді залежності зі стереотипом…
А. <<include>>
Б. <<incside>>
В. <<within>>
Г. <<contain>>
Д. <<switch on>>
23. Як на діаграмах прецедентів показується розширення прецедентів? В вигляді залежності зі стереотипом…
А.<<incside>>
Б.<<within>>
В. <<contain>>
Г. <<switch on>>
Д. <<extend>>
24. Які прецеденти називаються «включаючими прецедентами»?
А. Прецедент, який актори спостерігають при взаємодії з системою
Б. Дії, що виконуються сумісно декількома варіантами використання
В. Поведінка, що включається в деякий варіант використання
Г. Поведінка, що розширює функціонал, що надається варіантом використання
Д. Альтернативні варіанти поведінки системи, що визначаються деякою умовою
25. Які прецеденти називаються «узагальнюючиими прецедентами»?
А. Прецедент, який актори спостерігають при взаємодії з системою
Б. Прецедент, що містить дії, які виконуються сумісно декількома варіантами використання
В. Поведінка, що включається в деякий варіант використання
Г. Поведінка, що розширює функціонал, що надається варіантом використання
Д. Альтернативні варіанти поведінки системи, що визначаються деякою умовою

26. Які прецеденти називаються «розширюючими прецедентами» (extanded user case)?
А. Прецедент, який актори спостерігають при взаємодії з системою
Б. Дії, що виконуються сумісно декількома варіантами використання
В. Поведінка, що включається в деякий варіант використання
Г. Поведінка, що розширює функціонал, що надається варіантом використання
Д. Альтернативні варіанти поведінки системи, що визначаються деякою умовою
27. Які прецеденти називаються «повними прецедентами»?
А. Прецедент, який актори спостерігають при взаємодії з системою
Б. Дії, що виконуються сумісно декількома варіантами використання
В. Поведінка, що включається в деякий варіант використання
Г. Поведінка, що розширює функціонал, що надається варіантом використання
Д. Альтернативні варіанти поведінки системи, що визначаються деякою умовою
28. Що таке прецедент?
А. Функціональність, вже одного разу реалізована в деякій системі, на яку можна посилатися під час розробки
Б. Функціональність системи, що дозволяє користувачу отримати деякий значущий для нього, відчутний і вимірюваний результат
В. Функціональність системи, стандарт «де-факто» для систем подібного класу
Г. Функціональність системи, стандартна для всіх систем даного класу, реалізована в стандартних бібліотеках
Д. Функціональність системи, що дозволяє користувачу застосувати досвід, отриманий з іншими подібними системами
29. Виберіть зі списку НЕ істинне твердження, що стосуються прецедентів
А. Прецедент – це опис набору послідовних подій, що виконуються системою, які призводять до результату, який спостерігає актор
Б. Прецеденти описують сервіси, що надаються системою акторам, з якими вони взаємодіють
В. Прецедент ніколи не пояснює, «як» працює сервіс, а тільки описує, «що» робить
Г. Прецеденти зображують в вигляді класу зі стереотипом <<use case>>
Д. Ім’я прецеденту зазвичай набагато довше імен інших елементів
30. Який графічний символ слугує для зображення варіанта використання (use case) на діаграмі прецедентів?
А. Круг
Б. Еліпс
В. Прямокутник
Г. Ромб
Д. Пряма лінія
31. Яке з висловлювань є справедливим щодо відношення розширення (extend)?
А. Відношення розширення пов’язує актора з окремим варіантом використання
Б. Відношення розширення пов’язує три варіанта використання
В. Відношення розширення пов’язує лише два актора між собою
Г. Відношення розширення показує додаткові можливості базового варіанта використання або актора, що не є обов’язковими
Д. Відношення розширення пов’язує базовий варіант використання та варіант, що в нього обов’язково включається (входить)
32. Яке визначення сценарію вірне?
А. Сценарій являє собою вимоги до користувача, який взаємодії з системою
Б. Сценарій – це послідовність дій, що забезпечує досягнення системою заданої мети
В. Сценарій визначена послідовність дій, що описує дії акторів і поведінку модельованої системи в вигляді звичайного тексту
Г. Структура системи у графічному вигляді
Д. Сценарій – це опис виключних ситуацій, що доповнює діаграму прецедентів у вигляді звичайного тексту
33. Яке визначення конкретного класу вірне?
А. Це клас, що містить реалізацію своїх операцій
Б. Це клас, що має задані типи атрибутів і операцій
В. Це клас, на основі якого можуть бути безпосередньо створенні екземпляри (об’єкти)
Г. Це клас, від якого повинен наслідуватися клас, що має екземпляри (об’єкти)
Д. Це віртуальний клас
34. На якому етапі життєвого циклу програмного забезпечення створюється діаграма класів?

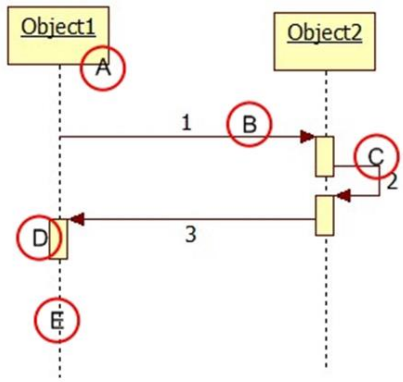
А. Підтримки
Б. Проектування
В. Реалізації
Г. Тестування
Д. Впровадження
35. Яким чином на діагармах UML зображується наслідування?
А. Не зафарбованою трикутною стрілкою, що направлена в сторону дочірнього класу
Б. Не зафарбованою трикутною стрілкою, що направлена в сторону батьківського класу
В. Не зафарбованою ромбовидною стрілкою, що направлена в сторону суперкласу
Г. Не зафарбованою ромбовидною стрілкою, що направлена в сторону підкласу
Д. Не зафарбованою двонаправленою трикутною стрілкою
36. Який тип асоціації називається n-нарною асоціацією
А. Це асоціація, що об’єднує три і більше класів
Б. Це асоціація, що об’єднує більше одного класу
В. Це асоціація з вказаною кратність на кінцях
Г. Це асоціація прецедента та актора
Д. Це асоціація між класом та суперкласом
37. Аналогом якої діаграми є діаграма кооперації (взаємодії)?
А. діаграма прецендентів
Б. діаграма станів
В. діаграма діяльності
Г. діаграма послідовності
Д. діаграма класів
38. Що означає символ ромб в діаграмі діяльності?



А. Злиття потоків діяльності
Б. Прийняття рішення
В. Розпарарелювання потоків діяльності
Г. Кінцевий стан
Д. Початковий стан
39. Що означає прямокутник на вертикальних лініях під об’єктом на діаграмах послідовностей?

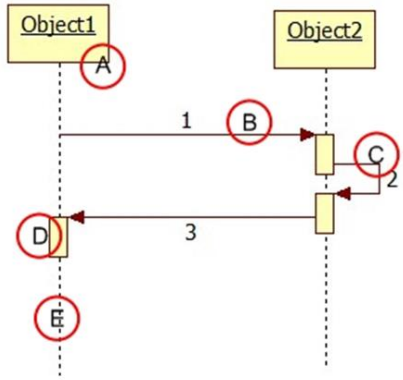


А. Фокус управління
Б. Створення та знищення об’єкту
В. Лінія життя об’єкту
Г. Отримання об’єктом інформації
Д. Передача інформаціх до іншого об’єкту
40. Якою буквою на малюнку зображено лінія життя об’єкта?



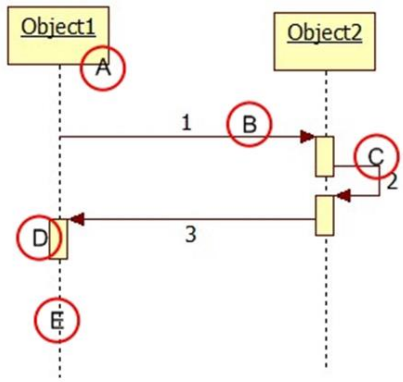
А. A
Б. B
В. C
Г. D
Д. E

41. Якою буквою на малюнку зображено фокус керування об’єкта?



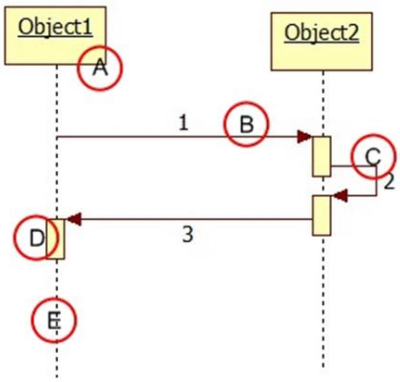
А. A
Б. B
В. C
Г. D
Д. E

42. Якою буквою на малюнку позначено відправлене повідомлення?



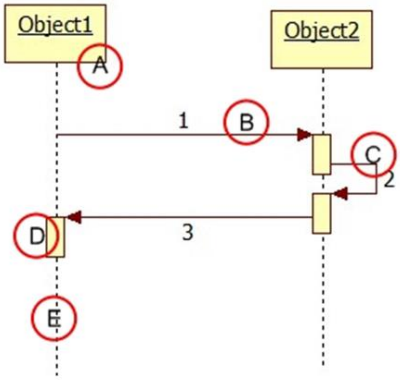
А. A
Б. B
В. C
Г. D
Д. E

43. Якою буквою на малюнку позначено об’єкт?



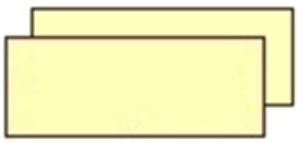
А. A
Б. B
В. C
Г. D
Д. E

44. Якою буквою на малюнку позначено рефлексивне повідомлення?



А. A
Б. B
В. C
Г. D
Д. E

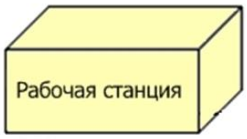
45. Який об’єкт діаграм кооперації зображено на малюнку?



А. Композитний об’єкт
Б. Частина композитного об’єкту
В. Порт
Г. Мультиоб’єкт
Д. Активний об’єкт

46. Як співвідносяться діаграми кооперації і діаграми об’єктів?
А. Діаграма об’єктів і діаграма кооперації повністю взаємозамінні
Б. Діаграма об’єктів і діаграма кооперації відрізняється лише нотацією
В. Використання діаграми об’єктів або діаграми кооперації залежить лише від уподобань розробника
Г. UML-модель не може містити діаграму об’єктів і діаграму кооперації одночасно
Д. Діаграма об’єктів показує статику, а діаграма взаємодії описує динамічні аспекти системи

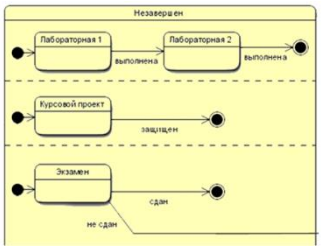
47. Що означають надписи під позначенням вузла?



UNIX
XWindow
appoint
Netscape
dia

А. Компоненти системи, що встановлюються на цей вузол
Б. Папки, відкриті для доступу на даному вузлі
В. Програмне забезпечення встановлене на даному вузлі
Г. Вимоги до апаратного та програмного забезпечення вузла
Д. Будь-які примітки, що стосуються даного вузла

48. Що означають символи станів, зображенні всередині великого символу стану і розділені пунктирними лініями?

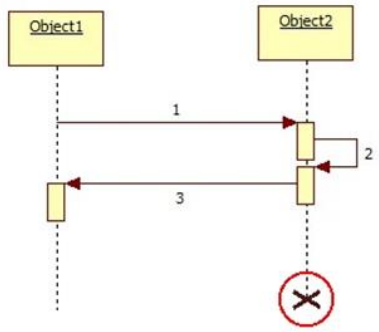


А. Паралельні підстани
Б. Стани активні у даний момент
В. Альтернативі підстани
Г. Варіанти деталізації стану
Д. Ієрархію станів

49. Що означає символ показаний на малюнку на діаграмі станів?

А. Злиття потоків діяльності
Б. Прийняття рішення
В. Розпаралелення потоків діяльності
Г. Кінцевий стан
Д. Початковий стан

50. Що означає символ, виділений на малюнку?



А. Знищення об’єкту
Б. Припинення передачі повідомлень
В. Виключення із взаємодії
Г. Відсутність фокуса керування
Д. Припинення приймання повідомлень

ЛІТЕРАТУРА.

1. Леоненков А.В. Самоучитель UML / А. В. Леоненков. − СПб. : БХВ-Петербург, 2004. − 432 с.
2. Леоненков А.В.Нотация и семантика языка UML. − Електронний ресурс. − http://www.intuit.ru/department/pl/umlbasics/
3. Бабич А. В. Введение в UML. − Електронний ресурс. − http://www.intuit.ru/department/se/intuml/
4. Д.В.Кознов Визуальное моделирование: теория и практика. − Електронний ресурс. − http://www.intuit.ru/department/se/vismodtp/
5. Г. Буч Объектно-ориентированный анализ и проектирование с примерами приложений на C++. 2-е изд.: Пер. с англ. − М.: Издательство Бином, СПб.: Невский диалект, 1999.
6. Г. Буч, Дж. Рамбо , А. Джекобсон Язык UML. Руководство пользователя.: Пер. с англ. − М.: ДМК, 2000. − 432с.
7. Брукс Ф. Мифический человеко-месяц или как создаются программные системы. — Пер. с англ. — СПб.: Символ-Плюс, 2001. — 304 с.
8. Глотова Т. В. Объектно-ориентированная методология разработки сложных систем. Учебное пособие. – 49 с.
9. Орлов С. Технологии разработки программного обеспечения: Учебник/ С. Орлов. — СПб.: Питер, 2002. — 464 с.: ил.
10. Кватрани Т. Визуальное моделирование с помощью Rational Rose 2002 и UML. − М.: Вильямс, 2003. − 192 с.

ДОДАТКИ

UML 2.0 прийнятий в якості міжнародного стандарту ISO / IEC 19501:2005. У UML використовуються наступні види діаграм (для виключення неоднозначності наведені також позначення англійською мовою):

Structure Diagrams:

Class diagram
Component diagram
Composite structure diagram
Collaboration (UML2.0)
Deployment diagram
Object diagram
Package diagram
Profile diagram (UML2.2)

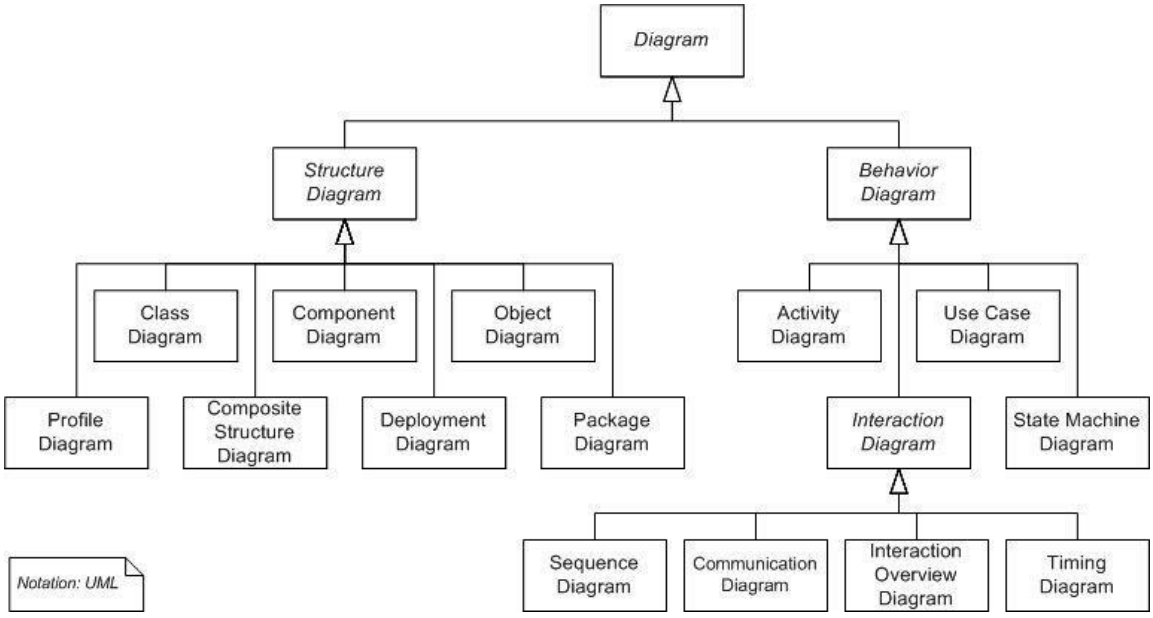
Behavior Diagrams:

Activity diagram
State Machine diagram Use case diagram
Interaction Diagrams:
Communication diagram (UML2.0) / Collaboration (UML1.x)
Interaction overview diagram (UML2.0)
Sequence diagram
Timing diagram (UML2.0)

Структурні діаграми:

- Класів
- Компонентів
- композитної / складовою структури о Кооперації (UML2.0)
- Розгортання
- Об'єктів
- Пакетів
- Профілів (UML2.2) Діаграми поведінки:
- Діяльності
- Станів
- Варіантів використання • Діаграми взаємодії: о Комунікації (UML2.0) / Кооперації (UML1.x) о Огляду взаємодії (UML2.0) о Послідовності о Синхронізації (UML2.0)

Структуру діаграм UML 2.3 можна представити на діаграмі класів UML:



Опис типів діаграм

1. Діаграма класів

Діаграма класів (Class diagram) - статична структурна діаграма, що описує структуру системи, вона демонструє класи системи, їх атрибути, методи і залежності між класами.

Існують різні точки зору на побудову діаграм класів залежно від цілей їх застосування:

 концептуальна точка зору - діаграма класів описує модель предметної області, в ній присутні лише класи прикладних об'єктів;

 точка зору специфікації - діаграма класів застосовується при проектуванні інформаційних систем;

 точка зору реалізації - діаграма класів містить класи, що використовуються безпосередньо в програмному коді (при використанні об'єктно-орієнтованих мов програмування).

2. Діаграма компонентів

Діаграма компонентів (Component diagram) - статична структурна діаграма, показує розбиття програмної системи на структурні компоненти та зв'язку (залежності) між компонентами. Як фізичних компонент можуть виступати файли, бібліотеки, модулі, виконувані файли, пакети тощо.

3. Діаграма композитної / складовою структури

Діаграма композитної / складовою структури (Composite structure diagram) - статична структурна діаграма, демонструє внутрішню структуру класів і, по можливості, взаємодію елементів (частин) внутрішньої структури класу.

Підвидом діаграм композитної структури є діаграми кооперації (Collaboration diagram, введені в UML 2.0), які показують ролі й взаємодію класів у рамках кооперації. Кооперації зручні при моделюванні шаблонів проектування.

Діаграми композитної структури можуть використовуватися спільно з діаграмами класів.

4. Діаграма розгортання

Діаграма розгортання (Deployment diagram) - служить для моделювання роботи вузлів (апаратних засобів, англ. Node) і артефактів, розгорнутих на них. У UML 2.X на вузлах розгортаються артефакти англ. artifact), в той час як в UML 1 на вузлах розгорталися компоненти. Між артефактом і логічним елементом (компонентом), який він реалізує, встановлюється залежність маніфестації.

5. Діаграма пакетів

Діаграма пакетів (Package diagram) - структурна діаграма, основним змістом якої є пакети і відносини між ними. Діаграми пакетів служать, в першу чергу, для організації елементів у групи за певною ознакою з метою спрощення структури та організації роботи з моделлю системи.

6. Діаграма діяльності

Діаграма діяльності (Activity diagram) - діаграма, на якій показано декомпозиція певної діяльності на її складові частини. Під діяльністю (англ. activity) розуміється специфікація поведінки у вигляді координованого послідовного і паралельного виконання підлеглих елементів - вкладених видів діяльності та окремих дій (англ. action), з'єднаних між собою потоками, які йдуть від виходів одного вузла до входів іншого. Діаграми діяльності використовуються при моделюванні бізнес-процесів, технологічних процесів, послідовних і паралельних обчислень.

7. Діаграма автомата / станів

Діаграма автомата (State Machine diagram, діаграма кінцевого автомата, діаграма станів) - діаграма, на якій представлений кінцевий автомат з простими станами, переходами і композитними станами.

Кінцевий автомат (англ. State machine) - специфікація послідовності станів, через які проходить об'єкт або взаємодія у відповідь на події свого життя, а також відповідні дії об'єкта на ці події. Кінцевий автомат прикріплений до вихідного елементу (класу, кооперації або методу) і служить для визначення поведінки його екземплярів.

8. Діаграма прецедентів

Діаграма прецедентів (Use case diagram, діаграма варіантів використання) - діаграма, на якій відображені відносини, що існують між акторами і прецедентами Основне завдання - засіб, що дає можливість замовнику, кінцевому користувачеві і розробнику спільно обговорювати функціональність і поведінку системи.

9. Діаграми комунікації та послідовності

Діаграми комунікації і послідовності транзитивні, описують взаємодію, але показують її різними способами і з достатнім ступенем точності можуть бути перетворені одна в іншу.

Діаграма комунікації (Communication diagram, в UML 1.x - діаграма кооперації, collaboration diagram) - діаграма, на якій зображується взаємодія між частинами композитної структури або ролями кооперації. На відміну від діаграми послідовності, на діаграмі комунікації явно вказуються відносини між елементами (об'єктами), а час як окремий вимір не використовується (застосовуються порядкові номери викликів).

Діаграма послідовності (Sequence diagram) - діаграма, на якій зображено впорядковану в часі взаємодію об'єктів. Зокрема, на ній зображуються об'єкти, що беруть участь у взаємодії і послідовність повідомлень, якими вони обмінюються.

10. Діаграма співпраці

Цей тип діаграм дозволяє описати взаємодію об'єктів, абстрагуючись від послідовності передачі повідомлень. На цьому типі діаграм в компактному вигляді відображаються всі прийняті і передані повідомлення конкретного об'єкта і типи цих повідомлень.

11. Діаграма взаємодії

Діаграма огляду взаємодії (Interaction overview diagram) - різновид діаграми діяльності, що включає фрагменти діаграми послідовності і конструкції потоку управління.

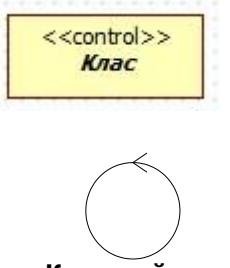
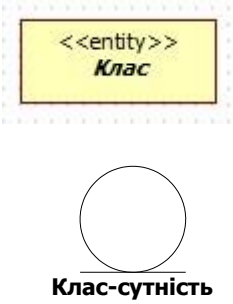
Цей тип діаграм включає в себе діаграми Sequence diagram (діаграми послідовностей дій) і Collaboration diagram (діаграми співробітництва). Ці діаграми дозволяють з різних точок зору розглянути взаємодію об'єктів у створюваній системі.

12. Діаграма синхронізації

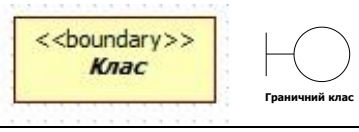
Діаграма синхронізації (Timing diagram) - альтернативне представлення діаграми послідовності, явним чином показує зміни стану на лінії життя із заданою шкалою часу. Може бути корисна в додатках реального часу.

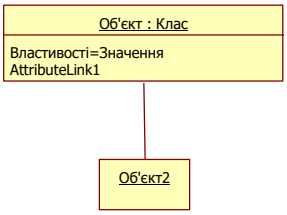
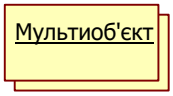
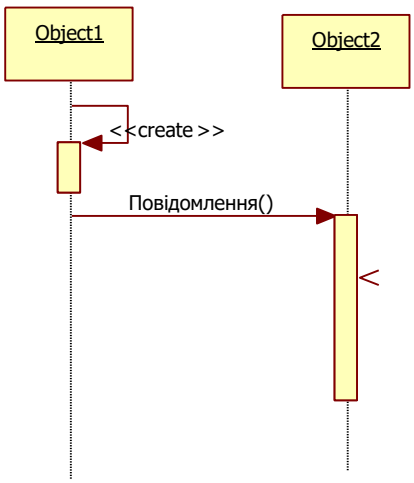
ЖДТУ	**Міністерство освіти і науки України** **Житомирський державний технологічний університет**

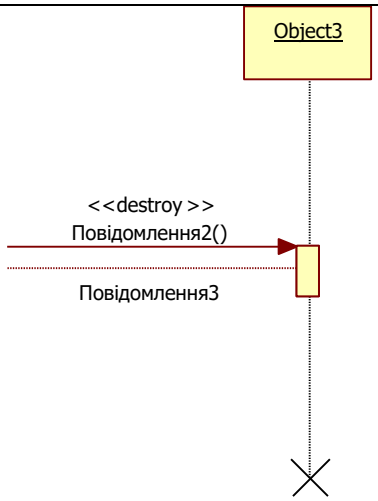
Основні позначення елементів діаграм

Диаграма прецедентів	
	Варіант використання являє собою специфікацію загальних особливостей поведінки або функціонування модельованої системи без розгляду внутрішньої структури цієї системи. Незважаючи на те, що кожен варіант використання визначає послідовність дій, які повинні бути виконані проектованої системою при взаємодії її з відповідним актором, самі ці дії не зображуються на розглянутій діаграмі.
	актор (actor) - узгоджена множина ролей, які мають зовнішні сутності по відношенню до варіантів використання при взаємодії з ними. Актор являє собою будь-яку зовнішню стосовно моделюється системі сутність, яка взаємодіє з системою і використовує її функціональні можливості для досягнення певної мети або вирішення приватних завдань. Кожен актор може розглядатися як якась окрема роль щодо конкретного варіанту використання.
Диаграмма класів	
	Клас (class) - абстрактне опис безлічі однорідних об'єктів, що мають однакові атрибути, операції і відносини з об'єктами інших класів.
	Керуючий клас (control class) - клас, що відповідає за координацію дій інших класів. На кожній діаграмі класів повинен бути хоча б один керуючий клас, причому кількість повідомлень, що надсилаються об'єктам керуючого класу менша, за число повідомлень, що розсилаються ними. Керуючий клас відповідає за координацію дій інших класів. Як правило, даний клас є активним і ініціює розсилку безлічі повідомлень інших класів моделі. Крім спеціального позначення керуючий клас може бути зображений у формі прямокутника класу із стереотипом << control >>
	Клас-сутність (entity class) - пасивний клас, інформація про який повинна зберігатися постійно і не знищуватися з виключенням системи. Клас-сутність містить інформацію, яка повинна зберігатися постійно і не знищується із знищенням об'єктів даного класу або припиненням роботи модельованої системи, пов'язані з виключенням системи або завершенням програми. Як правило, цей клас відповідає окремій таблиці бази даних. У цьому випадку його атрибути є полями таблиці, а операції - приєднаними або збереженими процедурами. Цей клас пасивний і лише приймає повідомлення від інших класів моделі. Клас - сутність може бути зображений також стандартним чином у формі прямокутника класу із стереотипом <<entity>>

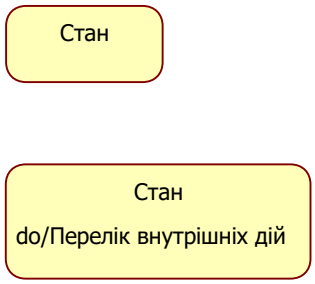
ЖДТУ	**Міністерство освіти і науки України** **Житомирський державний технологічний університет**

	Граничний клас (boundary class) - клас, який розташовується на кордоні системи із зовнішнім середовищем і безпосередньо взаємодіє з акторами, але є складовою частиною системи. Граничний клас може бути зображений також стандартним чином у формі прямокутника класу із стереотипом <<boundary>>

Діаграма кооперації	
Об'єкт : Клас Властивості=Значення	Об'єкт (object) - сутність з добре визначеними кордонами і індивідуальністю, яка інкапсулює стан і поведінку.
	Активний об'єкт (active object) має власний процес управління і може ініціювати діяльність з управління іншими об'єктами.
	Мультіоб'єкт (multiobject) являє собою безліч анонімних об'єктів, які можуть бути утворені на основі одного класу.
Діаграма послідовностей	
	Об'єкт графічно зображується у формі прямокутника і розташовується у верхній частині своєї лінії життя. Лінія життя об'єкта (object lifeline) - вертикальна лінія на діаграмі послідовності, яка представляє існування об'єкта протягом певного періоду часу. Лінія життя служить для позначення періоду часу, протягом якого об'єкт існує в системі і, отже, може потенційно брати участь у всіх її взаємодіях. Фокус управління (focus of control) - спеціальний символ на діаграмі послідовності, який вказує період часу, протягом якого об'єкт виконує деяку дію, перебуваючи в активному стані.

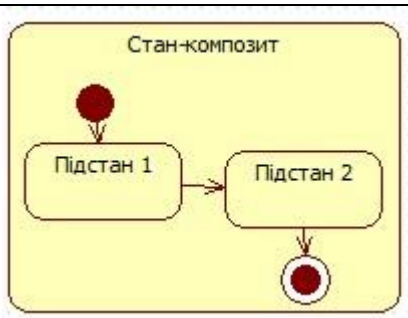


Діаграма станів



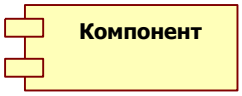
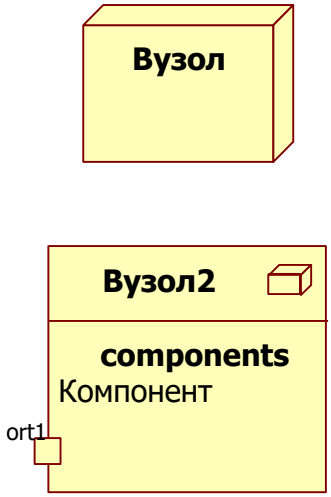
Стан (state) - умова або ситуація в ході життєвого циклу об'єкта, протягом якого він задовольняє логічного умові, виконує певну діяльність або очікує події.

Початковий стан (start state) - різновид псевдостану, що позначає початок виконання процесу зміни станів кінцевого автомата або знаходження модельованого об'єкта в складеному стані Кінцеве стан (final state) - різновид псевдостану, що позначає закінчення процесу зміни станів кінцевого автомата або знаходження модельованого об'єкта в складеному стані.



Складений стан (composite state) - складний стан, який складається з інших вкладених у нього станів.

Діаграма активностей

	Стан діяльності (activity state) - стан у графі діяльності, яке служить для представлення процедурної послідовності дій, що вимагають певного часу. Стан дії (action state) - спеціальний випадок стану з деякою вхідною дією і, принаймні, одним виходять зі стану переходом.
	Розгалуження - послідовно виконувана діяльність повинна розділитися наальтернативние гілки в залежності від значення проміжного результату. Така ситуація отримала назву розгалуження, а для її позначення застосовується спеціальний символ рішення.
Діаграма компонентів	
	Компонент (component). Компонент реалізує деякий набір інтерфейсів і служить для загального позначення елементів фізичного представлення моделі.
	інтерфейси забезпечують не тільки сумісність різних версій, але і можливість вносити істотні зміни в одні частини програми, не змінюючи інші її частини
Діаграма розгортання (deployment diagram) (діаграма топологій)	
	Вузол (node) є деяким фізично існуючим елементом системи, що володіє деяким обчислювальним ресурсом. Як обчислювальний ресурс вузла може розглядатися наявність щонайменше деякого об'єму електронної або магнітооптичної пам'яті та/або процесора. В останній версії UML поняття вузла розширено і може містити не тільки обчислювальні пристрої (процесори), але і інші механічні або електронні пристрої: принтери, модеми, цифрові камери, сканери і ін.