МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ
«ФАХОВИЙ КОЛЕДЖ ПРОМИСЛОВОЇ АВТОМАТИКИ
ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
ОДЕСЬКОГО НАЦІОНАЛЬНОГО ТЕХНОЛОГІЧНОГО УНІВЕРСИТЕТУ»

Циклова комісія комп'ютерних наук та інженерії програмного забезпечення

ЗАТВЕРДЖУЮ
директор ФКПАІТ ОНТУ
підписано Ольга ЄПУР
30.08.2023 року

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

обов'язкова

Освітньо-професійна програма Інженерія програмного забезпечення

Код та найменування спеціальності 121 «Інженерія програмного забезпечення»

Шифр та найменування галузі знань 12 «Інформаційні технології»

Мова навчання українська

2023

Розроблено та забезпечується: цикловою комісією Комп'ютерних наук та інженерії програмного забезпечення ВСП «Фаховий коледж промислової автоматики та інформаційних технологій Одеського національного технологічного університету»

Розробники:

- Тетяна КОСТИРЕНКО, викладач вищої кваліфікаційної категорії ФКПАІТ ОНТУ

Розглянуто та схвалено на засіданні циклової комісії Комп'ютерних наук та інженерії програмного забезпечення

Протокол №1 від 28.08.2023 р.

Голова циклової комісії *підписано* Тетяна КОСТИРЕНКО
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

Гарант освітньо-професійної програми *підписано* Тетяна КОСТИРЕНКО
(підпис) (Власне ім'я, ПРІЗВИЩЕ)

Розглянуто та схвалено Методичною радою ФКПАІТ ОНТУ

Протокол №1 від 29.08.2023 р.

Голова Методичної ради ФКПАІТ ОНТУ *підписано* Вікторія ОКСАНІЧЕНКО
(підпис)

1. Пояснювальна записка

Вступ

Програма вивчення навчальної дисципліни «Конструювання програмного забезпечення» складена відповідно до освітньо-професійної програми підготовки фахових молодших бакалаврів спеціальності 121 «Інженерія програмного забезпечення» та в стандарту фахової передвищої освіти із спеціальності 121 Інженерія програмного забезпечення.

Предмет вивчення

Предметом вивчення навчальної дисципліни є основи конструювання програмного забезпечення та уніфікована мова моделювання програмного забезпечення UML.

Міждисциплінарні зв'язки

Попередні – Основи програмної інженерії, Об'єктно-орієнтоване програмування, Бази даних, послідовні – WEB-програмування, Технології захисту інформації і кодування.

Мета та завдання навчальної дисципліни

Метою даної дисципліни є формування у здобувачів освіти компетенцій та навичок, необхідних для ефективного конструювання високоякісного програмного забезпечення з використанням сучасних методологій та інструментів розробки, з особливим акцентом на використання Unified Modeling Language (UML) для моделювання та документування.

Завдання:

- Оволодіння основами UML: здобувачі освіти повинні освоїти базові поняття та елементи UML, такі як діаграми варіантів використання, діаграми класів, діаграми послідовності тощо.
- Моделювання вимог за допомогою UML: вивчення та використання UML для моделювання вимог до програмного продукту. Створення діаграм варіантів використання для чіткого опису функціональності системи.
- Проектування системи за допомогою діаграм класів та співробітництва: засвоєння навичок проєктування системи через створення діаграм класів та діаграм співробітництва в UML. Визначення структури системи та взаємодії між компонентами.
- Моделювання послідовності дій: вивчення та використання діаграм послідовності для ілюстрації взаємодії між об'єктами системи в різних сценаріях використання.
- Моделювання станів: засвоєння навичок використання діаграм станів для представлення поведінки об'єктів та системи в різних станах.
- Використання UML для документування проєкту: навчання студентів використовувати UML як ефективний інструмент для документування проєктів, створення зрозумілих та повних технічних специфікацій.

- Робота над проєктом з використанням UML: важливою частиною дисципліни є реалізація здобутих знань у практичних проєктах, де студенти використовують UML для конструювання програмного забезпечення.

Метою цих завдань є навчання здобувачів освіти використовувати UML як ефективний інструмент для аналізу, проєктування та документування програмного забезпечення, що сприятиме їхній здатності ефективно співпрацювати та розробляти високоякісні програмні продукти у реальних умовах індустрії.

Компетентності та результати навчання

У результаті вивчення навчальної дисципліни «*Конструювання програмного забезпечення*» здобувач освіти отримує наступні програмні компетентності та програмні результати навчання, які визначені в Стандарті фахової передвищої освіти із спеціальності 121 Інженерія програмного забезпечення (https://mon.gov.ua/storage/app/media/vishcha-osvita/zatverdzeni%20standarty/12/21/121-inzhener.programn.zabezp.bakalavr-1.pdf) та освітньо-професійній програмі «Інженерія програмного забезпечення» (https://dev-kpa.fakel.com.ua/storage/uploads/4t0YvRV8MBZ1IXWV9i190ZBGF5H7rglXYysWLzuH.pdf) підготовки фахових молодших бакалаврів.

Загальні компетентності:

ЗК05. Знання та розуміння предметної області та розуміння професійної діяльності

ЗК06. Здатність до пошуку, оброблення та аналізу інформації з різних джерел

ЗК07. Здатність застосовувати знання у практичних ситуаціях

Спеціальні (фахові, предметні) компетентності:

СК02. Здатність накопичувати знання в галузі інформаційних технологій та усвідомлювати важливість навчання протягом усього життя.

СК04. Здатність дотримуватися стандартів при розробці програмного забезпечення.

СК05. Здатність брати участь у визначенні та формулюванні вимог до програмного продукту.

СК10. Здатність реалізовувати всі етапи життєвого циклу програмного забезпечення.

Програмні результати навчання:

РН02. Систематизувати та узагальнювати інформацію про підходи, методи та засоби розробки супроводу програмного забезпечення.

РН03. Застосовувати спеціалізовані емпіричні та теоретичні знання у сфері інженерії програмного забезпечення.

РН05. Розробляти та супроводжувати програмне забезпечення.

РН06. Використовувати основні методології та підходи до організації життєвого циклу програмного забезпечення.

РН08. Аналізувати вимоги до програмного забезпечення.

РН15. Аналізувати та узагальнювати необхідну інформацію з різних джерел та ресурсів для розв'язання професійних задач з урахуванням сучасних досягнень інформаційних технологій.

2. Інформаційний обсяг навчальної дисципліни

2.1 Тематичний план

Таблиця 2.1 – Тематичний план навчальної дисципліни

№	Назва змістових модулів і тем
Змістовий модуль 1. Організація процесу конструювання. Керівництво програмним проєктом	
Тема 1	Визначення технології конструювання програмного забезпечення. Стратегії конструювання.
Тема 2	Ваговиті і полегшені процеси
Тема 3	XP – процес
Тема 4	Процес керівництва проєктом
Тема 5	Планування проєктних задач
Тема 6	Виконання оцінки проєкта на основі LOC- та FP-метрик
Змістовий модуль 2 Базис мови візуального моделювання	
Тема 7	Уніфікована мова моделювання UML. Словник
Тема 8	Діаграми в UML. Механізми розширення.
Тема 9	Діаграми Use Case
Тема 10	Класи.
Тема 11	Діаграми співпраці. Діаграми послідовностей.
Тема 12	Моделювання поведінки програмної системи. Діаграма схем станів.
Тема 13	Діаграми діяльності.
Тема 14	Компонентні діаграми.
Тема 15	Діаграми розміщення
Змістовний модуль 3 *Взаємодія PHP з базою даних*	
Тема 16	Інтеграція. Тестування інтеграції
Тема 17	Основи шаблонів проєктування
Тема 18	Тестування «чорної скриньки». Тестування «білої скриньки»
Разом: 90 годин	

2.2 Зміст дисципліни

Змістовий модуль 1. Організація процесу конструювання. Керівництво програмним проєктом

Тема 1. Визначення технології конструювання програмного забезпечення. Стратегії конструювання

Розгляд терміну "технологія конструювання ПЗ" та його компонентів.

Критерії вибору технології: вимоги проекту, масштабність, швидкість розробки та інші. Основні стратегії конструювання: Інкрементальний та

ітеративний підходи, спіральна модель та модель каскаду. Роль правильного вибору технології та стратегії у успіху розробки.

Тема 2. Ваговиті і полегшені процеси в розробці програмного забезпечення

Ознайомитись з історією виникнення великовагових та полегшених процесів, розглянути їх основні недоліки. Аналіз переваг та обмежень ваговитих та полегшених підходів. Сценарії використання кожного підходу в різних ситуаціях. Подання прикладів успішного впровадження ваговитих та полегшених процесів у розробці програмного забезпечення

Тема 3. XP – процес

Ознайомитись з основною ідеєю Extreme Programming (XP)-проєктування. Розглянути методи екстремального проєктування, а також ознайомитись з моделями якості процесів конструювання.

Тема 4. Процес керівництва проєктом

Етапи керівництва програмним проєктом, виміри, заходи і метрики. Огляд ролі та функцій проєктного менеджера Визначення основних навичок та якостей, які роблять проєктного менеджера ефективним. Процес вибору та складання команди для конкретного проєкту. Важливість комунікації та створення позитивного робочого середовища.

Тема 5. Планування проєктних задач

Роль чіткої визначеності мети та цілей у плануванні. Взаємозв'язок мети та завдань проєкту. Методи інвентаризації та аналізу всіх задач проєкту. Створення ієрархії та структури для легкості управління. Використання методів, таких як Метрика SMART, для оцінки та формулювання задач. Планування проєктних задач. Розмінно-орієнтаційні метрики

Тема 6. Виконання оцінки проєкта на основі LOC- та FP-метрик

Мета оцінки, варіанти використання FP-даних, кроки виконання оцінки

Попередня оцінка програмного продукту

Детальний розгляд оцінки проєкта на основі LOC- та FP-метрик

Аналіз чутливості програмного проєкту

Детальний розбір аналізу чутливості програмного проєкту.

Змістовий модуль 2 Базис мови візуального моделювання

Тема 7. Уніфікована мова моделювання UML. Словник

Словник мови, предмети в UML, відношення та діаграми в UML..

Тема 8. Діаграми в UML. Механізми розширення.

Механізми розширення. Стереотип, тегова-величина, строка властивостей. Основна характеристика діаграм UML.

Тема 9. Діаграми Use Case

Основні поняття діаграми Use Case, прецеденти, актори, інтерфейси. Види відношень на діаграмі, графічна нотація діаграми прецедентів. Поняття діаграми, графічна нотація. правила побудови

Навики побудови діаграми за вибраною предметною областю.

Тема 10. Класи.

Загальна характеристика класів. Поняття атрибутів та операцій. Видимість атрибутів. Запис атрибутів в повній формі. Основні поняття діаграми класів, графічна нотація та правила побудови.

Навики виділення основних класів для обраної предметної області.

Графічна нотація та правила побудови діаграми класів

Тема 11. Діаграми співпраці. Діаграми послідовностей.

Основні поняття діаграми співпраці, графічна нотація діаграми.

Навики побудови діаграми за вибраною предметною областю.

Тема 12. Моделювання поведінки програмної системи. Діаграма схем станів.

Основні поняття діаграми станів, графічна нотація.

Навики побудови діаграми за вибраною предметною областю

Тема 13. Діаграми діяльності

Основні поняття діаграми діяльності, графічна нотація.

Навики побудови діаграми за вибраною предметною областю

Тема 14. Компонентні діаграми.

Основні поняття компонентної діаграми, інтерфейси, різновиди компонентів.

Основні поняття компонентної об’єктної моделі (COM).

Навики побудови діаграми за вибраною предметною областю.

Компонентні об’єктні системи. Різниця між COM об’єктом та звичайним об’єктом

Тема 15. Діаграми розміщення

Основні поняття діаграми розміщення, вузол, використання діаграм розміщення.

Навики побудови діаграми за вибраною предметною областю

Змістовний модуль 3 Якість конструювання. Шаблони проєктування

Тема 16. Інтеграція. Тестування інтеграції

Визначення інтеграції та її значення в контексті розробки програмного забезпечення. Відмінності між інтеграцією та юніт-тестуванням. Подання конкретних цілей та переваг, які можна досягти завдяки інтеграції. Оптимізація співпраці між різними частинами програмного продукту.

Тема 17. Основи шаблонів проєктування

Визначення та пояснення основних характеристик шаблонів проєктування. Роль шаблонів у полегшенні розробки та збільшенні повторного використання коду. Роль шаблонів у підвищенні якості, зрозумілості та обслуговуваності коду. Вплив шаблонів на архітектуру та структуру програмного забезпечення. Розділення шаблонів на структурні, поведінкові та породжувальні.

Тема 18. Тестування «чорної скриньки». Тестування «білої скриньки»

Огляд концепції тестування 'чорної скриньки'. Принципи тестування, не звертаючи уваги на внутрішню структуру програми. Розгляд різних методів та інструментів, використовуваних для проведення тестування 'чорної скриньки'. Переваги автоматизованого тестування та його вплив на процес. Огляд концепції тестування 'білої скриньки'. Зосередження на внутрішній

структурі програмного коду та логіці виконання. Розгляд методів, таких як юніт-тестування, інтеграційне тестування та інші, в рамках тестування 'білої скриньки'. Засоби для забезпечення високої якості коду та виявлення помилок.

2.3 Перелік практичних робіт з дисципліни

Таблиця 2.2 – Тематичний план практичних робіт

№	Тема практичної роботи
1.	Опис предметної області.
2.	Діаграми Use Case.
3.	Класи
4.	Асоціація між Агрегація. Композиція
5.	Побудова ієрархії простого спадкування. Залежність між класами
6.	Побудова діаграми класів.
7.	Діаграми співпраці. Діаграми послідовностей
8.	Моделювання поведінки програмної системи. Діаграма схем станів
9.	Діаграми діяльності
10.	Компонентні діаграми.
11.	Діаграми розміщення.
12.	Конструювання програмного забезпечення
	Всього: 30 годин

2.4 Кількість годин разом за програмою дисципліни

Розподіл дисципліни у годинах									
Курс	**I**		**II**		**III**		**IV**		**Всього**
Семестр	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	
Повний обсяг часу							90		90
Аудиторні заняття, годин							55		55
із них (кількість годин):									
лекції							25		25
практичні							30		30
Самостійна робота							35		35

3. Інструменти, обладнання та програмне забезпечення, використання яких передбачає дисципліна

1. Інструмент для побудови UML-схем: https://creately.com
2. Конструктор діаграм UML: https://www.lucidchart.com/pages/ru/examples/uml_diagram_tool

4. Інформаційні ресурси

Базові (основні):

1. Paul Hensgen «Umbrello UML Modeller», переклад Юрій Чорноіван, 2013. Доступ - https://docs.kde.org/trunk5/uk/umbrello/umbrello/index.html.
2. Крег Ларман, Застосування UML 2.0 та шаблонів проектування. 3-тє вид., Діалектіка, ISBN978-5-907144-36-1
3. Тихонов Є.С.. «Конструювання програмного забезпечення. Java». - 2018.
4. С. О. Цибульник, К. С. Барандич Технології розроблення програмного забезпечення частина 1. життєвий цикл програмного забезпечення, Київ КПІ ім. Ігоря Сікорського 2022

Додаткові:

1. 1. Дудзяний І.М. Об’єктно-орієнтоване моделювання програмних систем, Львів, видавничий центр, ЛНУ ім.. Івана Франка, 2007.
2. Лавріщева К.М. ПРОГРАМНА ІНЖЕНЕРІЯ.–К.– 2008.–319 с. ISBN 978–966–02–5052–9
3. Навчальний посібник з дисципліни «Технології розробки програмного забезпечення» для студентів спеціальності 123 «Комп’ютерна інженерія». – Полтава: ПолтНТУ, 2017. – 218 с.

5. Форма підсумкового контролю

Екзамен (VII семестр), поточний контроль.

6. Засоби діагностики результатів навчання

Перевірка та оцінювання знань здобувачів освіти може проводитись кількома методами:

1. Оцінювання знань здобувача освіти під час практичних занять.
2. Виконання індивідуальних навчально-дослідних завдань.
3. Захист практичних робіт.
4. Тестування.
5. Проведення поточно-модульного контролю.
6. Проведення екзамену.

6.1 Питання для самоконтролю

1. Які основні етапи розробки програмного забезпечення ви виділяєте при використанні UML?
2. Що таке UML і яку роль він відіграє у конструюванні програмного забезпечення?

3. Які діаграми UML ви вважаєте найбільш важливими для проєктування програмного забезпечення і чому?
4. Як можна використовувати діаграми взаємодії (Sequence Diagrams та Communication Diagrams) для моделювання взаємодії між об'єктами в системі?
5. Що таке діаграми класів UML, і як вони допомагають при моделюванні програмного забезпечення?
6. Як ви використовуєте UML для опису архітектури системи? Які діаграми ви застосовуєте?
7. Що таке UML діаграма станів і як вона може бути використана для моделювання поведінки об'єктів у системі?
8. Які інші види діаграм UML ви знаєте і в яких сценаріях їх можна застосовувати при конструюванні програмного забезпечення?
9. Як ви використовуєте UML для документування вимог до системи та їх зв'язку з архітектурою програмного забезпечення?
10. Які переваги та виклики пов'язані з використанням UML при конструюванні програмного забезпечення в командному середовищі?
11. Яким чином UML може бути використаний для моделювання взаємодії з зовнішніми системами та сервісами?
12. Як ви вирішуєте проблеми, пов'язані зі змінами вимог у середині процесу розробки за допомогою UML? Як забезпечуєте легкість модифікації системи?
13. Як використовуєте UML для покращення розуміння та комунікації з іншими членами вашої команди або зацікавленими стейкхолдерами?
14. Які інструменти для роботи з UML ви використовуєте та чому обрали саме їх?
15. Як UML може бути використаний для моделювання та аналізу ризиків у процесі розробки програмного забезпечення?
16. Як UML сприяє покращенню тестування та валідації програмного забезпечення?
17. Які основні етапи та кроки ви виконуєте при використанні UML для розробки програмного забезпечення?
18. Як ви забезпечуєте узгодженість між UML-моделями та фактичним кодом програмного забезпечення під час розробки?