Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра обчислювальної техніки

Розроблення проблемно-орієнтованих та сервісно-орієнтованих систем

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти *Другий (магістерський)*

Галузь знань	*12 Інформаційні технології*
Спеціальність	*121 Інженерія програмного забезпечення*
Освітня програма	*Інженерія програмного забезпечення комп'ютерних систем*
Статус дисципліни	*Нормативна*
Форма навчання	*очна(денна)*
Рік підготовки, семестр	*1 курс, осінній семестр*
Обсяг дисципліни	*6 кредитів*
Семестровий контроль/ контрольні заходи	*Екзамен*
Розклад занять	*Лекцій 36 годин, Лабораторних 36 годин, СРС 108 годин*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	Керівник: професор, д.т.н., Стіренко С.Г. sergii.stirenko@gmail.com *Старший викладач кафедри обчислювальної техніки* *Шемсединов Т.Г. timur.shemsedinov@gmail.com* *Асистент кафедри обчислювальної техніки* *Кухар В.В. kukhar.vitalii@gmail.com*
Розміщення курсу	https://github.com/HowProgrammingWorks

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчення та результати навчання

Метою навчальної дисципліни є формування у студентів здатностей (компетенцій) розробки програмного забезпечення для проблемно-орієнтованих (DSL) та сервісно-орієнтованих (SOA) систем. За результатами вивчення дисципліни здобувач має бути здатними вирішувати професійні завдання та володіти такими компетенціями:

- Здатність до абстрактного мислення, аналізу та синтезу(ЗК01);
- Здатність генерувати нові ідеї (креативність) (ЗК05);
- Здатність аналізувати предметні області, формувати, класифікувати вимоги до програмного забезпечення (ФК01);

- Здатність створювати та використовувати програмне забезпечення високопродуктивних комп'ютерних систем (ФК10);
- Здатність розробляти проблемно-орієнтовані та сервісно-орієнтовані системи (ФК11);
- Здатність використовувати мікросервісний підхід для створення програмних систем (ФК15)

Після засвоєння навчальної дисципліни здобувачі мають продемонструвати такі програмні результати навчання:

- Виявляти інформаційні потреби і класифікувати дані для проєктування програмного забезпечення (ПРН04);
- Розробляти і оцінювати стратегії проєктування програмних засобів обґрунтовувати, аналізувати і оцінювати варіанти проектних рішень з точки зору якості кінцевого програмного продукту, ресурсних обмежень та інших факторів (ПРН06).
- Розробляти і модифікувати архітектуру програмного забезпечення для реалізації вимог замовника (ПРН08);.
- обґрунтовано вибирати парадигми і мови програмування для розроблення програмного забезпечення; застосовувати на практиці сучасні засоби розроблення програмного забезпечення (ПРН09);
- збирати, аналізувати, оцінювати необхідну для розв'язання наукових і прикладних задач інформацію, використовуючи науково-технічну літературу, бази даних та інші джерела (ПРН17);
- Знати і застосовувати методи і технології створення проблемно-орієнтованих та сервісно-орієнтованих систем (ПРН18)

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Дисципліни, що передують: Програмування, Системне програмування, Паралельні та розподілені обчислення, Операційні системи, Гнучкі методи програмування, Моделювання програмного забезпечення.

Дана дисципліна є підготовкою до виконання курсової та дисертаційної магістерської робіт.

3. Зміст навчальної дисципліни

Тема 1. Архітектурний підхід до програмування: сервіси, модулі, порти та адаптери, компоненти.

Тема 2. Шари у DDD архітектурі, зв'язаність та зв'язність коду, принципи GRASP та SOLID у сервісах.

Тема 3. Сервісний підхід: програмні сервіси в середині додатку, веб-сервіси (SOA) та мікросервіси.

Тема 4. ServerlessClouds (FaaS) та ізоляція контекстів запитів.

Тема 5. Проектування API інтерфейсів та контрактів взаємодії систем.

Тема 6. Спеціалізовані мови для предметних областей - Domainspecificlanguages (DSL).

Тема 7. Кодогенерація, шаблони, використання узагальненого програмування для зміни поведінки.
Тема 8. Інтерпретація спеціалізованих мов, вбудовані мови, мови "склеювання" (glue).
Тема 9. Використання стандартних синтаксисів (ECMAScript, JSON, MD) для опису домену.
Тема 10. Підмножини та надмножини синтаксисів з готовою інфраструктурою та AST парсерами.
Тема 11. Зміни поведінки у Design-time, Compile-time, Run-time, Just-in-Time, Lazy.
Тема 12. Програмування на потоках даних та комутація сервісів.
Тема 13. Метапрограмування: прийоми та техніки, інтроспекція, рефлексія, скафолдінг.
Тема 14. Динамічна інтерпретація метамоделі.
Тема 15. Використання СКБД для гнучкого моделювання домену.
Тема 16. Засоби СКБД: ключ-значення, реляційна модель, ієрархічна модель та графові бази даних.
Тема 17. Безсхемні, об'єктно-орієнтовані та документо-орієнтовані бази даних.
Тема 18. Колонкові бази даних та in-memory бази даних, розподілені бази даних.
Тема 19. Високонавантажені розподілені програми узгоджені по даним.
Тема 20. Розподілені системи та CAP-теорема, Узгодженість, доступність та розподіленість.
Тема 21. Стратегії вирішення конфліктів та протоколи консенсусу.
Тема 22. Lock-free структури даних.
Тема 23. Безконфліктні репліковані типи даних (CRDT).

4. Навчальні матеріали та ресурси

Базова:

1. Шемсединов Т.Г., Нечай Д.О., Кухар В.В., Орленко О.А., Голіков О.Г., Білочуб М.М., Духін В., Іванова Л.А., Чорненький А.Ю. та інші. Приклади коду та приклади проектів [Електронний ресурс] розміщено за адресою: https://github.com/HowProgrammingWorks/
2. Domain-Specific Languages // Martin Fowler
3. Clean Code: A Handbook of Agile Software Craftsmanship // Robert C. Martin
4. Clean Architecture: A Craftsman's Guideto Software Structure and Design // Robert C. Martin
5. Gang of Four Design Patterns // Erich Gamma, Richard Helm, Ralph Johnson, andJohnVlissides
6. Analysis Patterns: Reusable Object Models // Martin Fowler
7. Domain-Driven Design: Tackling Complexity in the Hear tof Software 1st Edition // Eric Evans
8. Software Engineering Classics: Software Project Survival Guide/ Debugging the Development Process/ Dynamics of Software Development // Steve McConnell
9. Code Complete (Developer Best Practices) 2nd Edition // Steve McConnell
10. Designing Object Oriented C++ Applications Using The Booch Method // Robert C. Martin
11. Implementing Domain-Driven Design // VaughnVernon

Додаткова:

1. Шемсединов Т.Г., Нечай Д.О., Кухар В.В., Орленко О.А., Голіков О.Г., Білочуб М.М., Духін В., Іванова Л.А., Чорненький А.Ю. та інші. Технологічний стек Metarhia[Електронний ресурс] розміщено за адресою: https://github.com/metarhia/
2. Patterns of Enterprise Application Architecture // MartinFowler
3. More Effective Agile: A Roadmap for Software Leaders // Steve McConnell
4. Domain-Driven Design Reference: Definitions and Pattern Summaries // Eric Evans
5. Software Estimation: Demystifying the Black Art (Developer Best Practices) // Steve McConnell
6. Professional Software Development: Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers // Steve McConnell
7. Clean Agile: Backto Basics // Robert C. Martin
8. Agile Principles, Patterns, andPracticesin C# // Robert C. Martin
9. Agile Software Development, Principles, Patterns, and Practices // Robert C. Martin
10. The Mythical Man-Month: Essayson Software Engineering // Frederick Brooks
11. Extreme Programming Explained // KentBeck

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Основні завдання циклу лабораторних занять (комп'ютерного практикуму) -надбання практичних навичок створення формальних DSL (domain specific language) мов та опису метаданих для динамічного формування бізнес-процесів та доменної моделі, проектування інформаційних систем у архітектурі сервісів SOA (service-oriented architecture), використання DSL мов у SOA системах, використання готових інфраструктур для інтерпретації та виконання вбудованих DSL мов у розподілених сервісах.

№ з/п	Назва лабораторної роботи (комп'ютерного практикуму)	Кількість ауд. годин
1	Розробка архітектури інформаційної системи на базі сервісно-орієнтованої парадигми.	3
2	Аналіз предметної області (на свій вибір) з виділенням метаданих, які можна описати за допомогою DSL мови.	3
3	Проектування DSL мови для опису певної предметної області та створення прикладів використання мови.	3
4	Залучення готових синтаксичних аналізаторів, парсерів, інтерпретаторів, компіляторів, лінтерів та інших засобів розробки для обробки моделей у форматі DSL.	3
5	Використання вбудованих віртуальних машин, рантаймів, середовищ виконання у ІС на базі SOA.	3
6	Використання SQL та noSQL СКБД для розподілених SC на базі SOA та DSL.	3
	Разом:	18

6. Самостійна робота студента

У процесі виконання індивідуальних завдань студенти повинні опрацьовувати знання, отримані під час лекцій та самостійної роботи, самостійно вивчати визначені теми, поглиблювати свої знання для подальшого навчання. Самостійна робота студентів полягає в наступному:

- підготовці до лекційних занять по вивченню попереднього лекційного матеріалу;
- розглядання та модифікація прикладів коду та проектів з наданих викладачем gitрепозиторіїв;
- виконання лабораторних робіт з вивченням теорії та реалізацією наведеної теми у програмному коді;
- підготовка та участь у обговоренні тем на семінарах;
- пір-рев'ю роду колег студентів;
- підготовка та захищення коду на код-рев'ю викладача.

№ з/п	Назва теми, що виноситься на самостійне опрацювання	Кількість годин СРС
1	Створення SOA систем на базі хмарних FaaS сервісів.	4
2	Порівняння хмарних вендорів на предмет ізоляції контекстів при виконанні DSL мов.	4
3	Захист ІС на базі SOA та DSL від несанкціонованого втручання.	4

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Під час занять з навчальної дисципліни студенти повинні дотримуватись певних дисциплінарних правил:
1) так використовувати чат-групи, репозиторії та середовище виконання, щоб не створювати проблем з зайвими нотифікаціями іншим учасникам учбового процесу та викладачеві;
2) вчасно виконувати роботу та коммітити все кожного дня у систему контролю версій, не створювати багато коммітів у одному пул-реквесту, не створювати великіхкоммітів, розділяти все на окремі тематичні комміти, чітко та зрозуміло описувати комміти та пул-реквести, додавати теги для лінкуванняissues, PR, та коммітів у репозиторіях;
3) не допускаються використання піратських копій середовищ розробки, операційних систем чи інших засобів розробки та розгортання як на своїх обчислювальних засобах, так і на хмарних;
4)спамити, постити забагато мемів та стікерів у групах та репозиторіях;
5) якщо виникає питання, потрібно спочатку шукати у матеріалах курсу, потім у інтернеті, потім запитати у асистентів та інших студентів, а потім вже, тільки коли рішення не знайдено чи воно незрозуміле, турбувати викладача;

Лабораторні роботи здаються тільки через Github у відкритому коді (до коду студенти додають ліцензію MIT), а репозиторій повинен мати історію розробки, щоб викладач міг простежити послідовність написання коду та авторство. Розробка відбуваєтья у feature-

гілках git, після чого перевірка коду проходить через PR та включає рев'ю, яке викладач робить у пул-реквесті. Історія рев'ю залишається у Github аккаунті студента.

8. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Види контролю з навчальної дисципліни «Розроблення проблемно-орієнтованих та сервісно-орієнтованих систем» включають:

Лабораторні роботи

Заплановано самостійне виконання 6 лабораторних робіт.

Теми лабораторних робіт узгоджені у часі та за змістом з темами лекцій

Поточний контроль:

Передбачено 2 попередні рев'юкода за курс, які повністю охоплюють розглянуту на лекціях тематику навчальної дисципліни. До рев'ю коду викладач може додавати теоретичні питання, які розкривають розуміння теми студентом.

Семестровий контроль

Остаточний рев'ю коду робиться за декілька підходів, але по дедлайну пул-реквест зачиняється, та фіксуються усі зауваження, що виправлені чи не виправлені.

Екзамен

Проводиться у вигляді співбесіди зі студентом для об'єктивного визначення рівня знань, умінь та практичних навичок, отриманих за семестр чи парного лайвкодінгу за викладачем чи асистентом, парного кодінгу з іншим студентом чи повернення до написаного за семестр коду та усунення його недоліків чи обговорення його особливостей.

. Рейтинг студента з кредитного модуля складається з балів, які він отримує за види робіт відповідно до таблиці 4.

Таблиця 4

Оцінювання окремих видів навчальної роботи студента

Розділ 1,2		Розділ 3,4	
Вид навчальної роботи	Максимальна кількість балів	Вид навчальної роботи	Максимальна кількість балів
Виконання та захист лабораторної роботи № 1	10	Виконання та захист лабораторної роботи № 4	10
Виконання та захист лабораторної роботи № 2	10	Виконання та захист лабораторної роботи № 5	10
Виконання та захист лабораторної роботи № 3	10	Виконання та захист лабораторної роботи № 6	10
Поточний код-рев'ю №1	10	Поточний код-рев'ю №2	10
		Фінальний код-рев'ю	20
Усього за семестр			**100**

Разом за лабораторні роботи (максимальна кількість балів) – 60

Індивідуальний семестровий рейтинг студента (підсумкова семестрова рейтингова оцінка **RD**) є сумою балів, отриманих студентом протягом семестру за участі у семінарах, рев'ю коду, обговореннях.

Усі студенти, незалежно від того, чи вони виконали всі умови допуску до семестрової атестації з кредитного модуля та мають рейтингову оцінку не менше 60 балів, проходять співбесіду з викладачем.

Необхідною умовою допуску студента до екзамену є його індивідуальний семестровий рейтинг (**RD**) не менший, ніж 30% від максимуму балів, тобто 30 балів, здані 4 лабораторні роботи та наявність однієї позитивної атестації в семестрі. При невиконанні хоча б однієї зі згаданих умов студент до екзамену не допускається.

Сума підсумкової семестрової (**RD**) та екзаменаційної рейтингових оцінок у балах становить підсумкову семестрову рейтингову оцінку, яка перераховується в оцінки за національною шкалою та шкалою ECTS (табл. 5).

Таблиця 5

Відповідність рейтингових балів оцінкам за університетською шкалою

Кількість балів	*Оцінка*
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

Робочу програму навчальної дисципліни (силабус):

Склав професор кафедри обчислювальної техніки Стіренко С.Г.

Ухвалено кафедрою обчислювальної техніки (протокол № 10 від 25.05.2022)

Погоджено Методичною комісією факультету (протокол № 10 від 09.06.2022)