НАЦІОНАЛЬНИЙ ТРАНСПОРТНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ТРАНСПОРТНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ

ЗАТВЕРДЖУЮ:
завідувач кафедри інформаційних систем і технологій
проф. В.В. Гавриленко _____
_____ _____2020 р.

СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

ОКП15. АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Рівень вищої освіти	перший (бакалаврський)
Спеціальність	121 «Інженерія програмного забезпечення»
Освітня програма	«Інженерія програмного забезпечення»
Вид дисципліни	обов'язкова
Форма навчання	денна
Семестр	5-й семестр навчального плану
Викладач	ст. викладач Сватко Віталій Володимирович
Кафедра	інформаційних систем і технологій тел. кафедри: +38 (044) 280-70-66 веб-сайт кафедри: http://kist.ntu.edu.ua/
Доступ до матеріалів	Google Classroom: Software Architecture & Design (code X), Syllabus: http://kist.ntu.edu.ua/nmk_ipz_bak.php, Ел. підручники: http://kist.ntu.edu.ua/posib_ipz_bak.php
Об'яви	Google Classroom: Software Architecture & Design, ел. пошта групи, дошка оголошень і веб-сайт кафедри
Розклад на 2020-2021 н.р.	http://www.ntu.edu.ua/studentam/rozklad/

І. АНОТАЦІЯ

Метою вивчення дисципліни є отримання знань та навичок щодо опису внутрішньої структури програмного забезпечення, який стає основою для його побудови, командної розробки компонент архітектури та організації їх взаємодії за допомогою інтерфейсів.

Предметом вивчення дисципліни є методи й інструменти проектування й розробки архітектури програмного забезпечення.

Основними **завданнями** вивчення дисципліни є:

- вивчення основних принципів й методів проектування програмного забезпечення;
- вивчення основних типів архітектур програмного забезпечення;
- ознайомлення із практикою застосування шаблонів проектування;
- отримання навичок із проектування програмного забезпечення;
- отримання навичок із розробки й оцінки архітектури програмного забезпечення;
- отримання навичок демонстрації й реалізації розроблених проектів.

Мова викладання: українська, англійська.

І. ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Дисципліна включає два основних блоки: «Проектування програмного забезпечення» та «Архітектура програмного забезпечення».

В рамках блоку «Проектування» Ви дізнаєтеся, як створювати модульне і гнучке програмне забезпечення, застосовуючи принципи і рекомендації об'єктно-орієнтованого проектування. Ви зможете передавати ці проекти в візуальній нотації, відомій як уніфікована мова моделювання (UML). Вам буде запропоновано застосувати свої знання в області об'єктно-орієнтованого проектування, розвиваючи і документуючи кодову базу Java із відповідною документацією UML.

Вивчення шаблонів проектування створить основу для розробки більш складних програмних додатків. Нарешті, Ви зможете визначити проблемні програмні розробки, звернувшись до каталогу кодів «із запашком».

В рамках блоку «Архітектура» Ви вивчите способи подання організації взаємодії програмних компонент як в UML, так і в інших візуальних інструментах. Будуть представлені найбільш поширені архітектури, їх якості й альтернативи. Ми поговоримо про те, як оцінити архітектуру, яку архітектуру можна назвати «гарною» і як архітектура впливає на процес розробки програмного забезпечення. Ви будете давати оцінку запропонованої архітектури додатку з використанням методу порівняльного аналізу архітектури (Architecture Tradeoff Analysis Method).

Отже, засвоєння матеріалів лекцій, виконання й захист лабораторних робіт, розробка проекту програмного продукту в рамках курсової роботи згідно із програмою курсу дозволить сформувати **знання, уміння** й **навички**, необхідні для успішного працевлаштування й професійної діяльності фахівців із розробки програмного забезпечення.

Загальні й професійні компетентності

К01 – Здатність до абстрактного мислення, аналізу та синтезу.
К02 – Здатність застосовувати знання у практичних ситуаціях.
К06 – Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
К07 – Здатність працювати в команді.
К14 – Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.
К15 – Здатність розробляти архітектури, модулі та компоненти програмних систем.
К17 – Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу.
К19 – Володіння знаннями про інформаційні моделі даних, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних.
К20 – Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв’язання завдань інженерії програмного забезпечення.
К23 – Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.
К25 – Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення.
К26 – Здатність до алгоритмічного та логічного мислення.
К28 – Володіння знаннями про сучасні інформаційні технології навігації і зв’язку та їх застосування на транспорті.
К32 – Здатність проектувати програмне забезпечення, що входить до складу технології інтернету речей.

Програмні результати навчання

ПР05 – Знати і застосовувати відповідні математичні поняття, методи доменного, системного і *об’єктно-орієнтованого аналізу* та математичного моделювання для розробки програмного забезпечення.
ПР11 – Вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання.
ПР12 – Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.
ПР17 – Вміти застосовувати методи компонентної розробки програмного забезпечення.

Вимоги до студентів

Для успішного початку засвоєння матеріалу навчальної дисципліни необхідні знання й навички, отриманні з дисциплін «Професійна практика програмної інженерії», «Об'єктно-орієнтоване програмування» та «Основи програмної інженерії», уміння використовувати мову програмування Java на початковому рівні для створення простих проектів.

Зв'язок з іншими дисциплінами

У структурі начального плану курс з «Архітектури та проектування програмного забезпечення» логічно пов'язаний із дисциплінами «Аналіз вимог до програмного забезпечення», «Конструювання програмного забезпечення», «Моделювання та аналіз програмного забезпечення», «Якість програмного забезпечення та тестування» та практичними курсами вивчення різних мов програмування (схема 1).

Схема 1. Взаємозв'язок дисципліни «Архітектура та проектування програмного забезпечення» з іншими.

Розподіл навчальних годин за видами роботи студента напротязі семестру й сесії

Види робіт за навчальним планом	*Годин*
Аудиторні заняття, у т.ч.:	*48*
лекції	16
лабораторні роботи	32
Самостійна робота, у т.ч.:	*102*
підготовка до лабораторних робіт	74
підготовка до модульних контрольних робіт	6
виконання курсової роботи	20
підготовка до підсумкового контролю	2
Всього	*150*
Форма підсумкового контролю	*Екзамен, курсова робота*

Інструментарій та програмне середовище для виконання лабораторних робіт: UMLet, LucidChart, Miro, MS PowerPoint.

ІІІ. ЗАПЛАНОВАНІ ТЕМИ АУДИТОРНИХ ЗАНЯТЬ

Модуль І. Основні принципи проектування програмного забезпечення

Представлені тут концепції, нотації і термінологія складають основу для розуміння ролі і обсягу проектування програмного забезпечення. Ви повторюєте такі основні принципи проектування як абстрактне представлення, інкапсуляція й приховування інформації, декомпозиція (асоціація, агрегування, композиція) і модульність, узагальнення і наслідування, створення інтерфейсів, згуртованість й зв'язність, розподіл відповідальності та ін.. і як вони виражаються на Java та візуально передаються в UML.

Після встановлення початкових вимог до програмного забезпечення, процес проектування передбачає два основні напрямки діяльності: концептуальний проект та технічний проект. У цьому модулі ви зрозумієте важливість проектування та об'єктно-орієнтованого мислення та навчитесь проектувати програмне забезпечення за допомогою такого методу, як картки CRC.

Модуль ІІ. Шаблони проектування

В цьому модулі Ви знайомитеся зі структурою програмного забезпечення, а також найпоширенішими шаблонами проектування. Ви продовжуватимете вивчати та практикувати проектування у UML та кодувати деякі з цих шаблонів на Java. Крім цього, у випадку успішного засвоєння попереднього матеріалу додатково передбачається практика проектування інтерфейсу користувача для обраного типу застосунку.

Модуль ІІІ. Архітектура програмного забезпечення

У цьому модулі ви дізнаєтеся детальніше про архітектуру програмного забезпечення. Ви дізнаєтеся, чому архітектура важлива, що необхідно враховувати і як передавати архітектуру з використанням UML. Ми розглянемо різні архітектури, які доцільно вибрати для розробки свого програмного продукту.

Графік	Вид і тема аудиторного заняття	Години
Модуль І. Основні принципи проектування програмного забезпечення		
Тема 1. Основи проектування програмного забезпечення		
1 тиждень	Лекція №1. Загальні концепції проектування. Контекст проектування. Тести.	2
	Лабораторна робота №1. Проект User Story. Абстрактне представлення проекту за допомогою CRC карток.	2
2 тиждень	Лабораторна робота №2. Побудова діаграми класів на основі концептуального проекту. Визначення типів зв'язків під час декомпозиції.	2
3 тиждень	Лекція №2. Процес проектування програмного забезпечення.	2
	Лабораторна робота №3. Побудова діаграми послідовності UML для демонстрації замовникам взаємодії класів системи під час певної дії користувача системи.	2
4 тиждень	Лабораторна робота №4. Побудова діаграми станів UML на основі діаграми класів й діаграми послідовності.	2
5 тиждень	Лекція №3. Основні принципи розробки програмного забезпечення .SOLID принципи. Тести.	2
	Лабораторна робота №5. Побудова функціональної діаграми.	2
6 тиждень	Лабораторна робота №6. Побудова діаграм при структурному підході до програмування.	2
Тема 2. Основні проблеми проектування програмного забезпечення		
7 тиждень	Лекція №4. Основні проблеми проектування програмного забезпечення: паралельні процеси, обробка подій, забезпечення схоронності даних, розподіл компонентів, взаємодія та представлення, безпека.	2
	Лабораторна робота №6. Тести.	2
Всього		22
Модуль ІІ. Шаблони проектування		

Графік	Вид і тема аудиторного заняття	Години
Тема 3. Шаблони проектування		
8 тиждень	Лабораторна робота № 7. Використання шаблону Singleton.	2
9 тиждень	Лекція №5. Патерни (шаблони) проектування.	2
	Лабораторна робота № 8. Використання шаблону Factory method.	2
10 тиждень	Лабораторна робота № 9. Архітектурні шаблони.	2
Тема 4. Проектування інтерфейсу користувача (вибірково)		
11 тиждень	Лекція №6. Загальні принципи дизайну інтерфейсу користувача. Проблеми проектування інтерфейсу користувача. Проектування способів взаємодії користувачів. Проектування презентації інформації. Процес проектування інтерфейсу користувача. Локалізація та інтернаціоналізація. Метафори та концептуальні моделі.	2
	Лабораторна робота № 10 Проектування інтерфейсу користувача системи. Тести.	2
Всього		12
Модуль ІІІ. Архітектура програмного забезпечення		
Тема 5. Процес розробки архітектури програмного забезпечення		
12 тиждень	Лабораторна робота № 11 Case studies. Побудова логічної, фізичної й технічної діаграм для відображення архітектури застосунків.	2
13 тиждень	Лекція №7. Процес розробки архітектури компонент різних типів застосунків.	2
	Лабораторна робота №12. Архітектура програмної системи для моніторингу показників роботи безпілотного автомобіля	2
14 тиждень	Лабораторна робота №13. Метод порівняльного аналізу архітектури (Architecture Tradeoff Analysis Method)	2
Тема 6. Архітектурні стилі		
15 тиждень	Лекція №8. Архітектурні конструкції. Прийняття рішення щодо архітектури програмного забезпечення. Лінійка програмних продуктів.	2
	Лабораторна робота №14. Архітектура для IoT.	2
16 тиждень	Тести. Захист курсової роботи	2
Всього		14
За семестр		48

Для виконання лабораторної роботи передбачається 2-3 години самостійної підготовки студента, для повторення навчального матеріалу на модульну контрольну роботу – 2 години.

Курсова робота

Етапи виконання роботи	*Тижнів*	*Кількість годин*
Вибір напряму дослідження	1	1
Пошук і опрацювання літератури	1	2
Формулювання та затвердження теми	1	1
Складання плану курсової роботи	1	1
Розробка проекту	8	10
Формування тексту роботи і списку літератури	2	3
Оформлення пояснювальної записки та захист	2	2
Всього	*16*	*20*

Рекомендована література для самостійної підготовки

Основна література:

1. Опорний конспект лекцій (у вигляді тексту й презентацій) з дисципліни, НТУ, 2019 р.
   Джерело доступу: Google Classroom 'Software Design & Architecture'
2. Kevin Zhang. *Design Patterns. Elements of Reusable Object-Oriented Software*. 1997.
   Джерело доступу: http://www.uml.org.cn/c++/pdf/designpatterns.pdf
3. Давыдовский М.А., Никольская М.Н. *Проектирование программной системы в UML Designer.* Учебное пособие.- М.: Российский университет транспорта (МИИТ), 2019. -129 с.
   Джерело доступу:
   http://library.miit.ru/methodics/28062019/%D0%94%D0%B0%D0%B2%D1%8B%D0%B4%D0%BE%D0%B2%D1%81%D0%BA%D0%B8%D0%B9.pdf
4. Форд Нил, Парсонс Ребекка, Куа Патрик. *Эволюционная архитектура. Поддержка непрерывных изменений*. - СПб.: Питер, 2019. - 272 с.: ил. - (Серия «Бестселлеры O'Reilly»).
   Джерело доступу: https://webbooks.com.ua/download/?key=&count=1947&cat=29
5. D. Budgen.*Software Design*, 2nd ed., Addison-Wesley, 2003. - 489 p.
   Джерело доступу: http://www.dim.uchile.cl/~juaperez/beto/otro.bueno.pdf
6. L. Bass, P. Clements, and R. Kazman.*Software Architecture in Practice*, 3rd ed.,Addison-Wesley Professional, 2013.
   Джерело доступу: https://jegadeesansite.files.wordpress.com/2018/01/sei-series-in-software-engineering-len-bass-paul-clements-rick-kazman-software-architecture-in-practice-addison-wesley-professional-2012.pdf
7. Muhammad Ali Babar, Alan W. Brown, IvanMistrik.*Agile Software Architecture***,** Newnes, 2014. - 404 p. - ISBN 0124078850, 978-0124078857.
   Джерело доступу: http://index-of.co.uk/Agile/Agile%20Software%20Architecture%20(2014).pdf
8. Табунщик Г. В. Проектування та моделювання програмного забезпечення сучасних інформаційних систем / Г. В. Табунщик, Т.І. Каплієнко, О.А. Петрова – Запоріжжя :Дике Поле, 2016. – 250 с.
   Джерело доступу: http://eir.zntu.edu.ua/bitstream/123456789/1824/1/Tabunshchik_Software_Design.pdf
9. P.B. Kruchten, "The 4+1 View Model of Architecture," *IEEE Software*, vol. 12, no.6, 1995, pp. 42–55.

Додаткова література, рекомендована IEEE Computer Society (Digital Library: https://www.computer.org/csdl/home):

1. *IEEE Std. 12207-2008 (a.k.a. ISO/IEC12207:2008) Standard for Systems and Software Engineering—Software Life CycleProcesses*, IEEE, 2008.
2. T. DeMarco, "The Paradox of Software Architecture and Design," Stevens Prize Lecture, 1999.
3. D. Budgen, *Software Design*, 2nd ed.,Addison-Wesley, 2003.
4. I. Sommerville, *Software Engineering*, 9thed., Addison-Wesley, 2011.
5. M. Page-Jones, *Fundamentals of Object-Oriented Design in UML*, 1st ed., Addison-Wesley, 1999.
6. *IEEE Std. 1069-2009 Standard for Information Technology—Systems Design—Software Design Descriptions*, IEEE, 2009.
7. *ISO/IEC 42010:2011 Systems and Software Engineering—Recommended Practice for Architectural Description of Software-Intensive Systems*, ISO/IEC, 2011.
8. J. Bosch, *Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach*, ACM Press, 2000.
9. G. Kiczales et al., "Aspect-Oriented Programming," *Proc. 11th European Conf. Object-Oriented Programming* (ECOOP97), Springer, 1997.
10. J.G. Brookshear, *Computer Science: An Overview*, 10th ed., Addison-Wesley, 2008.
11. J.H. Allen et al., *Software Security Engineering: A Guide for Project Managers*, Addison-Wesley, 2008.
12. P. Clements et al., *Documenting Software Architectures: Views and Beyond*, 2nd ed., Pearson Education, 2010.
13. E. Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*, 1st ed., Addison-Wesley Professional, 1994.
14. I. Jacobson, G. Booch, and J. Rumbaugh,*The Unified Software Development Process*, Addison-Wesley Professional, 1999.
15. J. Nielsen, *Usability Engineering*, Morgan Kaufmann, 1993.
16. G. Booch, J. Rumbaugh, and I. Jacobson, The Unified Modeling Language User Guide, Addison-Wesley, 1999.
17. R.S. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed., McGraw- Hill, 2010.

IV. ПОРЯДОК ТА КРИТЕРІЇ ОЦІНЮВАННЯ

Форми контролю

Поточна форма контролю: захист лабораторних робіт, виконання практичних завдань під час аудиторних занять, тести.

Форма модульного контролю: тести.

Підсумкова форма контрою: екзамен, курсова робота.

Розподіл балів за формами контролю:

Тести	20%
Практичні завдання	10%
Лабораторні роботи	30%
Екзамен	40%
	100%
Курсова робота	100%

Виконання й захист всіх лабораторних робіт є обов'язковою умовою для допуску студента до екзамену, оскільки формує у студента більшість програмних компетентностей й результатів навчання в рамках цієї дисципліни. Тести дають можливість оцінити Ваші знання із переліку наведених вище програмних результатів. Практичні завдання під час аудиторних занять формують здатність працювати у команді. В рамках курсової роботи Ви навчаєтесь застосовувати на практиці інструментальні програмні засоби проектування, візуалізації та документування програмного забезпечення.

Порядок проведення екзамену включає письмову роботу студента з відповідями на теоретичні питання й одну прикладну задачу, а також, за необхідності, співбесіду студента із викладачем/комісією. Порядок оскарження рішення екзаменатора щодо оцінки визначений у «Положенні про освітній процес НТУ». Для дистанційного проведення занять передбачено комп'ютерне тестування.

Якщо максимальна кількість балів за кожне теоретичне питання – 10 балів, тоді:

– за повністю розкриту відповідь на питання студент одержує 10 балів;

– якщо студент дав відповідь на питання, однак допустив незначні помилки, він одержує 7 балів;

– якщо у відповіді не повністю розкрито сутність питання та допущені невірні тлумачення, студент одержує 3 бали;

– якщо студент не надав відповідь на питання, не виконав завдання, або виконав завдання з принциповими помилками, він одержує 0 балів.

Практичне завдання оцінюється у 20 балів. Кожний правильний крок у його вирішенні додає до загальної оцінки по 2 бали.

Підсумкову семестрову рейтингову оцінку в балах, за національною шкалою та шкалою ECTS вносять до заліково-екзаменаційної відомості, навчальної картки, індивідуального плану та залікової книжки студента.

Шкала оцінювання

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою
		для екзамену, курсової роботи
90 – 100	A	відмінно
82-89	B	добре
74-81	C	
64-73	D	задовільно
60-63	E	
35-59	FX	незадовільно з можливістю повторного складання
0-34	F	незадовільно з обов'язковим повторним вивченням дисципліни

V. ПОЛІТИКА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Відвідування є обов'язковим (за винятком випадків, коли існує поважна причина, наприклад, хвороба чи дозвіл працівників деканату). Якщо Ви не можете бути присутніми на заняттях, Ви все одно несете відповідальність за виконання завдань, що проводились в комп'ютерному класі, а також маєте дізнатись про всі оголошення. Студенти, які мають більше ніж 30% необґрунтованих пропусків, отримають остаточну оцінку, знижену на повний бал (за національною шкалою); студентів, загальна кількість пропусків яких (виправдані та невиправдані) перевищує 80%, з невиконаними лабораторними роботами й незахищеною курсовою роботою на екзамен не допускають.

Якщо Ви не маєте можливість вчасно показати й захистити індивідуальне завдання на аудиторних заняттях, це можна зробити пізніше під час консультацій, з графіком яких ознайомтесь завчасно на кафедрі інформаційних систем і технологій.

Можливо, Вам доведеться користуватися ноутбуками протягом певної частини навчального процесу. Під час обговорення практичних завдань, будь ласка, не використовуйте свої ноутбуки, смартфони, планшети чи комп'ютери в аудиторії. Це може відволікати викладача і студентів у групі, а також заважати зосереджуватися на матеріалі. Якщо ви використовуєте свій ноутбук чи телефон для аудіо чи відеозапису, необхідно заздалегідь отримати дозвіл викладача. **Повага** один до одного дає можливість ефективніше досягати поставлених командних результатів.

Всі індивідуальні завдання, заплановані у даній дисципліні, студент має виконати самостійно із використанням рекомендованої літератури й отриманих знань та навичок. Цитування в письмових роботах допускається тільки із відповідним посиланням на авторський текст. Питання **академічної доброчесності** студентів й викладачів регламентується Положенням про систему забезпечення академічної доброчесності педагогічними, науково-педагогічними, науковими працівниками та здобувачами вищої освіти в Національному транспортному університеті, Кодексом етики академічних взаємовідносин та доброчесності Національного транспортного університету

Викладач залишає за собою право під час навчального процесу змінювати, за необхідності й з обов'язковим попередженням про це студентів, послідовність викладання тем, графік консультацій, політику, розподіл балів, викладені в цьому силабусі.

1 вересня 2020