

**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ
КАФЕДРА ІНТЕЛЕКТУАЛЬНИХ ПРОГРАМНИХ СИСТЕМ**

«ЗАТВЕРДЖУЮ»

Заступник декана
з навчальної роботи

_____ Кашпур О.Ф.

«__» _____ 2019 року

**РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
МОДЕЛЕ-ОРІЄНТОВАНА ПОБУДОВА
ПРОГРАМНИХ СИСТЕМ
для студентів**

галузь знань	12 Інформаційні технології
спеціальність	121 Інженерія програмного забезпечення
освітній рівень	магістр
освітня програма	Програмне забезпечення систем
вид дисципліни	обов'язкова

Форма навчання	денна
Навчальний рік	2019/2020
Семестр	4
Кількість кредитів ECTS	4
Мова викладання, навчання та оцінювання	українська
Форма заключного контролю	іспит

Викладач: **д. ф.-м. н., професор Єршов С.В.** (лекції).

Пролонговано: на 20__/20__ н. р. _____ (_____) «__»__ 20__ р.

на 20__/20__ н. р. _____ (_____) «__»__ 20__ р.

КИЇВ – 2019

Розробник: Єршов Сергій Володимирович, д. ф.-м. н., с. н. с, професор кафедри інтелектуальних програмних систем.

ЗАТВЕРДЖЕНО

Завідувач кафедри інтелектуальних програмних систем

_____ О.І. Провотар

Протокол № __ від «__» _____ 2019 р.

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики

Протокол від «__» _____ 2019 року №__

Голова науково-методичної комісії _____ Л.Л. Омельчук

«__» _____ 2019 року

Затверджено вченою радою факультету комп'ютерних наук та кібернетики

Протокол від «__» _____ 2019 року №__

Голова вченої ради факультету _____ А.В. Анісімов

1. Мета дисципліни – ознайомлення студентів з актуальними практичними підходами до моделі-орієнтованої розробки програмного забезпечення.

2. Попередні вимоги до опанування навчальної дисципліни: відсутні.

3. Анотація навчальної дисципліни. Навчальна дисципліна «Моделі-орієнтована побудова програмних систем» є складовою освітньо-наукової програми підготовки фахівців за другим (магістерським) рівнем вищої освіти у галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення в рамках освітньо-наукової програми «Програмне забезпечення систем».

Дана дисципліна є нормативною навчальною дисципліною в рамках освітньої програми «Програмне забезпечення систем». Викладається у 4 семестрі в обсязі – **120 год. (4 кредити ECTS)**, зокрема: лекції – 30 год., самостійна робота – 88 год., консультації – 2 год. У курсі передбачено 2 змістовні частини. Завершується дисципліна – **іспитом**.

В рамках вивчення дисципліни розглядаються: основні поняття моделі-орієнтованої архітектури програмних систем; методи розробки предметно-орієнтованих мов моделювання; процес розробки метамоделей; методи та засоби трансформації моделей програмного забезпечення (ПЗ); інструменти та методи для автоматизації моделі-орієнтованого реінжинірингу і міграції (перенесення) програмних систем на нові перспективні платформи.

Дисципліна «Моделі-орієнтована побудова програмних систем» базується на знання і навичках, отриманих при вивченні дисципліни «Теоретичні основи та методи розробки інформаційних систем» другого (магістерського) рівня вищої освіти у галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення, в рамках освітньо-наукової програми «Програмне забезпечення систем».

4. Завдання (навчальні цілі). Основними завданнями дисципліни «Моделі-орієнтована побудова програмних систем» є набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень в області побудови та трансформації програмних систем відповідно до освітньої кваліфікації магістр з інженерії програмного забезпечення. Зокрема, розвивати:

- Здатність до абстрактного мислення, аналізу та синтезу (ЗК-1).
- Здатність проведення теоретичних та прикладних досліджень на відповідному рівні (ЗК-3).
- Здатність удосконалювати свої навички на основі аналізу попереднього досвіду (ЗК-6).
- Здатність генерувати нові ідеї (креативність) (ЗК-7).
- Здатність аналізувати предметні області, формувати, аналізувати та моделювати вимоги до програмного забезпечення (СК-1).
- Здатність проектувати програмне забезпечення, включаючи проведення моделювання його архітектури, поведінки та процесів функціонування окремих підсистем і модулів (СК-3).
- Вміння планувати і проводити наукові дослідження, готувати результати наукових робіт з інженерії програмного забезпечення до оприлюднення (СК-9).
- Здатність застосовувати і розвивати фундаментальні і міждисциплінарні знання для успішного розв'язання наукових завдань інженерії програмного забезпечення (СК-10).

5. Результати навчання за дисципліною.

Результат навчання (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		Форми (та/або методи і технології) викладання і навчання	Методи оцінювання та пороговий критерій оцінювання (за необхідності)	Відсоток у підсумковій оцінці з дисципліни
Код	Результат навчання			
РН1.1	Знати принципи моделювання орієнтованої архітектури програмного забезпечення.	Лекції.	Перевірка реферування літератури з тематики курсу, іспит	12%
РН1.2	Знати принципи розробки мов моделювання та трансформації ПЗ.	Лекції.	Перевірка реферування літератури з тематики курсу, іспит	10%
РН1.3	Знати основні методи для автоматизації моделювання орієнтованого реінжинірингу і міграції (перенесення) програмних систем на нові перспективні платформи.	Лекції.	Перевірка реферування літератури з тематики курсу, іспит	10%
РН2.1	Вміти використовувати методи і прийоми моделювання орієнтованої розробки для поширених типів ПЗ.	Самостійна робота.	Перевірка оцінки якості, тестування та пробна експлуатація ПЗ, іспит	20%
РН2.2	Вміти самостійно здійснювати трансформації моделей ПЗ як вручну, так і за допомогою сучасних інструментальних засобів автоматизації трансформацій.	Самостійна робота.	Перевірка програмної реалізації прототипу ПЗ, оцінки якості, тестування та пробна експлуатація ПЗ	20%
РН3.1	Обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань моделей ПЗ та процесу розробки метамodelей, скласти письмові звіти.	Самостійна робота.	Презентація теми проекту, розробка технічного завдання, розробка метамodelей ПЗ	15%
РН4.1	Організувати свою самостійну роботу для досягнення результату.	Самостійна робота.	Перевірка програмної реалізації прототипу ПЗ	8%
РН4.2	Відповідально ставитися до виконуваних робіт, нести відповідальність за їх якість.	Самостійна робота.	Перевірка презентації теми проекту, технічного завдання, метамodelей ПЗ	5%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання.

Результати навчання дисципліни	РН1.1	РН1.2	РН1.3	РН2.1	РН2.2	РН3.1	РН4.1	РН4.2
	Програмні результати навчання							
ПРН-1. Знати і системно застосовувати методи аналізу та моделювання прикладної області, виявлення інформаційних потреб і збору вихідних даних для проектування програмного забезпечення	+	+				+	+	+
ПРН-3. Знати і застосовувати базові концепції і методології моделювання інформаційних процесів	+		+					
ПРН-4. Оцінювати і обирати методи і моделі розробки, впровадження, експлуатації програмних засобів та управління ними на всіх етапах життєвого циклу.			+					+
ПРН-6. Аналізувати, оцінювати і обирати методи, сучасні програмно-апаратні інструментальні та обчислювальні засоби, технології, алгоритмічні та програмні рішення для ефективного виконання конкретних виробничих задач з програмної інженерії			+	+		+		
ПРН-8. Проводити аналітичне дослідження параметрів функціонування програмних систем для їх валідації та верифікації, а також проводити аналіз обраних методів, засобів автоматизованого проектування та реалізації програмного забезпечення			+	+	+		+	
ПРН-11. Набувати нові наукові і професійні знання, вдосконалювати навички, прогнозувати розвиток програмних систем та інформаційних технологій.					+	+	+	
ПРН-12. Формулювати, експериментально підтверджувати, обґрунтовувати і застосовувати на практиці в процесі розробки програмного забезпечення конкурентоспроможні ідеї, методи, технології вирішення професійних, науково-технічних завдань в умовах невизначеності.	+	+		+			+	+
ПРН-13. Оформляти результати досліджень у вигляді статей у наукових виданнях та тез доповідей на науково-технічних конференціях.				+	+		+	+
ПРН-14. Пояснити, аналізувати, цілеспрямовано шукати і обирати необхідні для вирішення фахових наукових і прикладних задач інформаційно-довідкові та науково-технічні ресурси і джерела знань з урахуванням сучасних досягнень науки і техніки.				+	+		+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів.

Семестрове оцінювання:

1. Презентація теми проекту: РН3.1., РН4.2 – **5 балів/3 бали.**
2. Розробка технічного завдання: РН3.1, РН4.2 – **5 балів/3 бали.**
3. Розробка метамоделі ПЗ: РН3.1, РН4.2 – **5 балів/3 бали.**
4. Реферування літератури з тематики курсу: РН1.1, РН1.2, РН1.3 – **10 балів/6 балів.**
5. Програмна реалізація прототипу ПЗ: РН2.2, РН4.1 – **25 балів/15 балів.**
6. Оцінка якості, тестування та пробна експлуатація ПЗ: РН2.1, РН2.2 – **10 балів/6 балів.**

Підсумкове оцінювання (у формі іспиту):

- Максимальна кількість балів які можуть бути отримані студентом: 40 балів.
- Результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН1.3, РН2.1.
- Форма проведення і види завдань: письмова робота.
- Види завдань: 20 тестових завдань.

Критерії оцінювання на іспиті.

Завдання	Тема завдання	Вага складових у відсотках
Завдання 1-3	Моделе-орієнтована архітектура програмних систем.	16%
Завдання 4-7	Object Constraint Language (Мова об'єктних обмежень).	16%
Завдання 8-12	Методи розробки мов моделювання.	20%
Завдання 13-15	Розробка метамоделей.	14%
Завдання 16-18	Трансформація моделей програмного забезпечення.	14%
Завдання 19	Трансформації «модель-текст».	10%
Завдання 20	Моделе-орієнтований реінжиніринг програмних систем.	10%
		100%

Запитання для підготовки до іспиту.

1. Основні принципи моделе-орієнтованої архітектури програмних систем.
2. Основні принципи предметно-орієнтованих мов.
3. Способи розробки мов моделювання.
4. Операції на основі ітераторів мови OCL (Object Constraint Language, мова об'єктних обмежень).
5. Операції щодо колекції мови OCL.
6. Запит інформації в моделі мовою OCL.
7. Побудова виразів мовою OCL.
8. Типи мови OCL. Навести приклади.
9. Моделе-орієнтована розробка на основі Object Constraint Language.

10. Способи використання мови OCL.
11. Приклади побудови предметно-орієнтованих мов моделювання.
12. Внутрішні та зовнішні предметно-орієнтовані мови.
13. Декларативні та імперативні, візуальні та текстові предметно-орієнтовані мови.
14. Класифікація предметно-орієнтованих мов за спрямуванням.
15. Основні принципи предметно-орієнтованих мов.
16. Засоби розширення мови UML (Unified Modeling Language, уніфікована мова моделювання).
17. Основні принципи моделі-орієнтованої архітектури.
18. Структурні діаграми UML (статичні діаграми).
19. Основні поняття мов моделювання.
20. Об'єкт моделі-орієнтованої розробки програмних систем.
21. Поведінкові динамічні діаграми UML.
22. Внутрішні та зовнішні предметно-орієнтовані мови.
23. Стереотипи мови UML.
24. Типи даних мови UML.
25. Моделі для конкретних платформ в MDA (Model-Driven Architecture, моделі-орієнтована архітектура).
26. Рівні моделювання в MDA.
27. Незалежні від платформи моделі в MDA.
28. Принципи метамоделювання.
29. Особливості розширення моделі при низхідному підході до моделі-орієнтованої міграції.
30. Види конкретного синтаксису мов моделювання.
31. Абстракція на основі API при здійсненні моделі-орієнтованої міграції.
32. Розробка конкретного синтаксису мов моделювання.
33. Особливості ендогенних графових трансформацій.
34. Особливості асоціації та композиції в MOF (Meta object Facility).
35. Основні ознаки для класифікації програмної системи як застарілої (успадкованої).
36. Використання Xtext для побудови конкретного синтаксису.
37. Формальні основи екстремального моделі-орієнтованого проектування XMDD.
38. Абстрактний синтаксис мов моделювання.
39. Типізовані атрибутивні графи у графових трансформаціях.
40. Запит інформації в моделі мовою OCL.
41. Розробка мов моделювання на основі метамоделей.
42. Моделі-орієнтована міграція програмних систем.
43. Мета-об'єктні засоби MOF (Meta object Facility).
44. Негативні умови застосування при трансформації графів.
45. Правила трансформації графів.
46. Типи мови MOF (Meta object Facility).
47. Незалежний вид платформи формат для обміну даними при моделі-орієнтованому реінжинірингу.
48. Відповідність між розширеною формою Бекуса-Наура та MOF.
49. Формальне визначення трансформації графів.
50. Ітераційний процес розробки метамоделей.
51. Особливості генерації коду при трансформації «модель-текст».
52. Приклад розробки метамоделей для діаграм діяльності.
53. Обробка коду програми аналізатором в процесі моделі-орієнтованого реінжинірингу.

54. Фази виконання трансформацій в ATL (ATLAS Transformation Language, мова трансформацій ATLAS).
55. Генерація коду на основі засобів мов програмування.
56. Ендогенні стратегії трансформації моделей. Навести приклади.
57. Характеристики мов трансформації «модель-текст».
58. Графічний конкретний синтаксис.
59. Основні поняття мови трансформації Acceleo.
60. Основні підходи до розробки графічного конкретного синтаксис.
61. Графічне представлення НСБ (незалежних від сервісу блоків) при моделі-орієнтованому реінжинірингу.
62. Побудова графічного конкретного синтаксису на основі відображення.
63. Основні ознаки для класифікації програмної системи як застарілої (успадкованої).
64. Побудова графічного конкретного синтаксису на основі анотації. Навести приклад.
65. З'ясування з існуючим кодом при низхідному підході до моделі-орієнтованої міграції.
66. Основні підходи до розробки текстового конкретного синтаксису.
67. Особливості моделі-орієнтованого реінжинірингу.
68. Огляд архітектури Xtext.
69. Моделі-орієнтована міграція програмних систем.
70. Архітектура трансформацій «модель-модель».
71. Низхідний підхід до моделі-орієнтованої міграції програмних систем.
72. Підготовка вихідного коду до моделі-орієнтованого реінжинірингу.
73. Екзогенні стратегії трансформації моделей. Навести приклади.
74. Ендогенні стратегії трансформації моделей. Навести приклади.
75. Структурна основа моделей коду при моделі-орієнтованому реінжинірингу.
76. Засоби мови ATL для здійснення трансформації моделей.
77. Приклад реконструкції застосунку при здійсненні моделі-орієнтованої міграції.
78. Фази виконання трансформацій в ATL.
79. Формальне визначення трансформації графів.
80. Спадкування правил мови ATL.
81. Характеристики мов трансформації «модель-текст».
82. Ітераційний процес розробки метамodelей.
83. Абстракція на основі API при здійсненні моделі-орієнтованої міграції.
84. Мета-об'єктні засоби MOF (Meta object Facility).
85. Опис процесу абстрагування при здійсненні моделі-орієнтованої міграції.
86. Вихідні та цільові шаблони у правилах мови ATL.
87. Особливості генерації коду при трансформаціях «модель-текст».

Студенти не допускаються до екзамену, якщо під час семестру вони набрали менше ніж 36 балів.

7.2 Організація оцінювання.

Терміни проведення форм оцінювання:

1. Презентація теми проекту: до 2 тижня семестру.
2. Розробка технічного завдання: до 2 тижня семестру.
3. Розробка метамodelі ПЗ: до 3 тижня семестру.
4. Реферування літератури з тематики курсу: на протязі семестру (визначається індивідуально, за згодою сторін, але не пізніше 3 тижня).
5. Програмна реалізація прототипу ПЗ: до 4 тижня семестру.
6. Оцінка якості та пробна експлуатація ПЗ: до 4 тижня семестру.

Студент має право на одне перескладання кожної роботи із можливістю отримання максимально 80% початково визначених за цю роботу балів. Термін перескладання визначається викладачем.

У випадку відсутності студента з поважних причин відпрацювання та перездачі контрольних робіт здійснюються у відповідності до „Положення про порядок оцінювання знань студентів при кредитно-модульній системі організації навчального процесу” від 1 жовтня 2010 року.

У разі неякісного виконання визначених вище робіт, викладач має право не зарахувати роботу, або знизити за неї бали.

Студент має право здавати роботи після закінчення визначеного для них терміну, але з втратою 20% від максимальної оцінки за кожен тиждень, який пройшов з моменту закінчення терміну її здачі.

7.3 Шкала відповідності оцінок.

Відмінно / Excellent	90-100
Добре / Good	75-89
Задовільно / Satisfactory	60-74
Незадовільно / Fail	0-59

8. Структура навчальної дисципліни. Тематичний план лекцій.

№ лекції	Назва лекції	Кількість годин	
		Лекції	Самостійна робота
Частина 1. Основні поняття та методи моделі-орієнтованої побудови програмного забезпечення.			
1	Тема 1. Принципи моделі-орієнтованої побудови програмного забезпечення.	2	2
2	Тема 2. Моделі-орієнтована архітектура програмних систем.	2	6
3	Тема 3. Основні поняття мов моделювання.	2	4
4	Тема 4. Object Constraint Language (Мова об'єктних обмежень).	2	8
5	Тема 5. Методи розробки мов моделювання.	2	6
6	Тема 6. Процес розробки метамodelей.	2	6
7	Тема 7. Розробка конкретного синтаксису для метамodelей.	2	8
8	Тема 8. Текстовий конкретний синтаксис мов моделювання.	2	6
Всього по частині 1		16	46

Частина 2. Методи та засоби трансформації програмних систем та міграції на інші платформи.			
9	Тема 9. Трансформації «модель-модель».	2	4
10	Тема 10. Ендогенні та графові трансформації моделей.	2	4
11	Тема 11. Трансформації програмного забезпечення «модель-текст».	2	8
12	Тема 12. Методи моделі-орієнтованої міграції програмного забезпечення.	2	6
13	Тема 13. Низхідний підхід до моделі-орієнтованої міграції.	2	6
14	Тема 14. Моделі-орієнтований реінжиніринг програмних систем.	2	6
15	Тема 15. Приклади моделі-орієнтованої міграції програмного забезпечення.	2	8
Всього по частині 2		14	42
Консультація		2	
ВСЬОГО		30	88

Загальний обсяг – **120** год., в тому числі:

Лекції – **30** год.

Самостійна робота – **88** год.

Консультації – **2** год.

9. Рекомендовані джерела.

Основні:

1. Brambilla M., Cabot J., Wimmer M. Model-Driven Software Engineering in Practice, 2nd Edition. – Morgan & Claypool Publishers, 2017. – 280 p.
2. Weilkiens T., Lamm J. G., Roth S., Walker M. Model-Based System Architecture. – Wiley, 2015. – 314 p.
3. Bettini L. Implementing domain-specific languages with Xtext and Xtend. Second Edition. – Packt Publishing, 2016. – 426 p. – ISBN:978-1-78646-496-5.
4. Wagner C. Model-Driven Software Migration: A Methodology: Reengineering, Recovery and Modernization of Legacy Systems. – Vieweg+Teubner Verlag, 2014. – 293 p.
5. Фаулер М. Предметно-ориентированные языки программирования. – М.: Издательский дом Вильямс, 2011. – 576 с.
6. Ларман К. Применение UML 2.0 и шаблонов проектирования, 3-е издание. – М.: Издательский дом Вильямс, 2019. – 736 с.
7. Wazlawick R.S. Object-Oriented Analysis and Design for Information Systems Modeling with UML, OCL, and IFML. – N.Y.: Elsevier, 2014. – 489 p. – ISBN: 0124186734.

Додаткові:

1. Frankel D. Model Driven Architecture: Applying MDA to Enterprise Computing. Willey, 2003. – 352 p.
2. Stahl T., Völter M. Model-Driven Software Development: Technology, Engineering, Management. – John Wiley & Sons, 2006. – 420 p. – ISBN-13: 978-0-470-02570-3.
3. Грибачев, К. Г. Delphi и Model Driven Architecture. Разработка приложений баз данных / К. Г. Грибачев. – СПб.: Питер, 2004. – 348 с.
4. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений (UML 2). Третье издание / Гради Буч, Роберт А. Максимчук, Майкл У. Энгл, Бобби Дж. Янг, Джим Коаллен, Келли А. Хьюстон. – М.: Издательский дом Вильямс, 2010. – 720 с.
5. Рамбо, Дж. UML 2.0. Объектно-ориентированное моделирование и разработка / Дж. Рамбо. – СПб.: Питер, 2007. – 544 с.
6. Вон Вернон. Реализация методов предметно-ориентированного проектирования (DDD). – М.: Издательский дом Вильямс, 2016. – 688 с.
7. Мацяшек Л.А. Анализ и проектирование информационных систем с помощью UML 2.0. Третье издание. – М.: Издательский дом Вильямс, 2008. – 816 с.
8. Völter M. DSL Engineering: Designing, Implementing and Using Domain-Specific Languages. – CreateSpace Independent Publishing Platform, 2013. – 558 p. – ISBN-13: 978-1481218580.