МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО

МЕТОДИЧНІ ВКАЗІВКИ
ЩОДО ВИКОНАННЯ САМОСТІЙНОЇ РОБОТИ
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ»
ДЛЯ СТУДЕНТІВ ДЕННОЇ ФОРМИ НАВЧАННЯ
ЗІ СПЕЦІАЛЬНОСТІ 123 – «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»
(ЧАСТИНА І)

КРЕМЕНЧУК 2020

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Інженерія програмного забезпечення» (частина І) для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія»

Укладач к. т. н., доц. О. Г. Славко

Рецензент к. т. н., доц. В. М. Сидоренко

Кафедра комп'ютерних та інформаційних систем

Затверджено методичною радою КрНУ імені Михайла Остроградського
Протокол № __ від _____2020 р.

Голова методичної ради _____ проф. В. В. Костін

ЗМІСТ

ВСТУП

Навчальна дисципліна «Інженерія програмного забезпечення» є логічним продовженням навчальних дисциплін «Проектування інформаційних систем», «Системи керування базами даних», «Комп'ютерні системи», «Прикладне програмування в комп'ютерних мережах» та ін. Її мета – набуття студентами загальних знань у галузі програмної інженерії на основі інженерних методів і технологій проектування програмного забезпечення.

Предметом вивчення навчальної дисципліни є основні технології та методи розробки програмного забезпечення, програмних проектів і критичних систем.

Програмний проект – це процес створення складного програмного продукту, що виконується великим колективом висококваліфікованих фахівців.

Отримання навичок практичної роботи з інструментальними засобами проектування програмних проектів і використання сучасних програмних засобів для проектування та реалізації програмного забезпечення є актуальним і складним завданням.

Мета та завдання навчальної дисципліни: набуття студентами загальних теоретичних і практичних знань у галузі програмної інженерії, що базуються на сучасних інженерних методах і технологіях проектування надійних, якісних програмних проектів, зокрема для систем реального часу, а також організації функціонування та супровід програмних проектів і тестування розроблюваного програмного забезпечення.

Місце навчальної дисципліни у навчальному процесі: навчальна дисципліна базується на знаннях з навчальних дисциплін «Проектування інформаційних систем», «Комп'ютерні мережі», «Комп'ютерні системи», «Програмування», «Організація баз даних», «Системи керування базами даних» та ін.

У результаті вивчення навчальної дисципліни студент повинен

знати:

– основні сучасні інструменти програмної інженерії;

– основні принципи побудови програм і методології програмування;

– процес розробки програмного забезпечення: універсального процесу, поточного процесу, стандартного процесу, удосконалення процесу, Push/Pull стратегії;

– структуру класичної моделі процесу розробки програмного забезпечення;

– архітектуру програмного забезпечення;

– основи керування вимогами: види, властивості та формалізація вимог;

уміти:

– створювати діаграми класів;

– розробляти структуру та взаємозв’язки проекту;

– розробляти документацію до проекту;

– працювати з декількома версіями проекту;

– проектувати системи критичного значення та системи реального часу;

– забезпечувати безпеку та стабільність проектованих критичних систем.

1 ТЕМИ ТА ПОГОДИННИЙ РОЗКЛАД ЛЕКЦІЙ І САМОСТІЙНОЇ РОБОТИ

Назви змістових модулів і тем	Кількість годин				
	Денна форма				
	Усього	У тому числі			
		л	п	лаб	с.р.
Модуль 1 Моделі розробки і реалізації програмних систем					
Змістовий модуль 1 Моделі ЖЦ для розробки програмних систем					
Тема 1 Програмна інженерія як фах. Базові поняття програмної інженерії	10	2			8
Тема 2 Загальні вимоги до програмного забезпечення. Керування вимогами	12	2		2	8
Тема 3 Процеси життєвого циклу (ЖЦ) ПЗ	12	2		2	8
Тема 4 Керування проектами	12	2		2	8
Усього за змістовим модулем 1	46	8		6	32
Змістовий модуль 2 Реалізація об’єктно орієнтованих програмних систем					
Тема 5 Основи об’єктно-орієнтованого подання програмних систем	12	2		2	8
Тема 6 Візуальне моделювання	14	2		2	10
Тема 7 Динамічні моделі об’єктно орієнтованих програмних систем	16	2		4	10
Тема 8 Моделі реалізації об’єктно орієнтованих програмних систем	18	4		4	10
Усього за змістовим модулем 2	64	10	-	18	38
ІНДЗ (КР, РГ, к/р)	-	-	-	-	-
Семестровий контроль (іспит)	4	-	-	-	4
Усього годин	110	18	-	18	74

2 ЗАВДАННЯ ДЛЯ САМОСТІЙНОЇ РОБОТИ

Тема. Планування робіт з виконання проекту «Розробка і впровадження програмного забезпечення»

Мета: набуття практичних навичок планування робіт з розробки та впровадження автоматизованих інформаційних систем на основі поширених моделей життєвих циклів програмних продуктів.

Короткі теоретичні відомості

Для проведення успішного проекту потрібно оцінити обсяг майбутніх робіт, можливий ризик, необхідні ресурси, майбутні завдання, визначити контрольні точки, вартість і план робіт, яким бажано користуватися. Процес керування програмним проектом охоплює розв'язання названих вище завдань. Цей процес починається перед технічною роботою, триває в міру розвитку програмного забезпечення (ПЗ) від ідеї до реальності та досягає найбільшої інтенсивності до кінця робіт.

Основним завданням під час планування є визначення структури розподілу робіт WBS (Work Breakdown Structure) за допомогою засобів планування робіт (наприклад, MS Project).

План виконання робіт складається на підставі декомпозиції проекту аж до постановки елементарних завдань, які можуть бути з'ясовані за результатами попереднього аналізу.

До того ж, можливе застосування змістовних моделей системного аналізу. Наприклад, використання моделі декомпозиції типу «життєвий цикл» дозволить поділяти окремі завдання на під завдання, визначаючи послідовність дій.

Для кожного елементарного завдання мають бути визначені:

– ресурси, необхідні для розв'язання завдання (у тому числі трудові);

– обсяг робіт, поданий у прийнятій системі метрик;

– вартість робіт (може бути обчислена за обсягом робіт і вартістю залучених ресурсів);

– тривалість робіт (може бути обчислена за обсягом робіт, кількістю залучених трудових ресурсів і прийнятих нормативів продуктивності).

Між окремими елементарними завданнями можуть бути певні залежності, які полягають у тому, що одні завдання можуть розв'язуватися паралельно, інші – у суворій послідовності (для розв'язання одних завдань можуть вимагатися результати виконання інших).

Після визначення залежностей можна розпочинати визначення елементарних завдань за часом. Для цього особливо слід зосередитися на завданнях, які розв'язуються паралельно.

Якщо план передбачає використання ресурсу 1 для розв'язання завдань А і Б, то ці завдання не можуть розв'язуватись паралельно, навіть якщо між ними немає концептуальної залежності. Окрім того, керівник проекту повинен знати завдання, що знаходяться на критичному шляху. Для того, щоб увесь проект був виконаний у строк, необхідно розв'язувати у строк усі критичні завдання.

Насамперед визначте основні фази виконання проекту. В основу може бути покладена прийнята модель життєвого циклу процесу розробки програмного забезпечення.

Наприклад, з використанням каскадної моделі основними фазами є аналіз, проектування, реалізація, тестування, упровадження. Потім визначте склад робіт усередині кожної фази, відповідно до суті розробки. Так буде визначено склад робіт за проектом. Кожна фаза має закінчуватися віхою – спеціальною одиницею роботи, що передбачає контроль виконання робіт за проектом і досягнення деякого проміжного або остаточного результату.

Потім визначте тривалість кожної роботи за планом робіт. Для визначення тривалості можуть бути використані різні регресійні моделі (наприклад, COCOMO II), або може застосовуватися прямий метод оцінювання.

Наступним етапом є визначення зв'язків між завданнями. У більшості засобів планування (наприклад, у MS Project), існує чотири види зв'язків.

Зв’язок типу *закінчення–початок* означає, що завдання В не може початися раніше, ніж закінчиться завдання А (наприклад, якщо під час розв’язання завдання Б використовуються результати, напівочікувані під час розв’язання завдання А).

Зв’язок типу *початок–початок* означає, що завдання В не може розпочатися до тих пір, поки не почалося розв’язання завдання А.

Наприклад, тестування програмного компонента не може починатися до того, як була розпочата його розробка, але, водночас, для написання тестів не обов’язково чекати завершення розробки цього компонента.

Зв’язок типу *закінчення–закінчення* означає, що робота Б не може закінчитися до тих пір, поки не завершиться виконання роботи А. Наприклад, проектування бази даних не може бути закінчено до того, як буде завершено семантичне моделювання предметної області.

Зв’язок *закінчення–початок* означає, що завдання Б не може закінчитися до того, як почалося завдання А. Такий зв’язок використовується, коли А є завданням з фіксованою датою початку, яку не можна змінити.

Після того, як визначено склад робіт, потрібно визначити, хто ці завдання розв’язуватиме і яке обладнання слід використовувати.

Сформуйте список ресурсів, для кожного ресурсу визначте назву та вартість його використання. Призначте ресурси для розв’язання конкретних завдань. З першим призначенням ресурсу будуть автоматично розраховані трудовитрати. Якщо необхідно прискорити розв’язання тих чи інших завдань, для цього можуть бути призначені додаткові ресурси.

Після розподілу ресурсів необхідно виконати вирівнювання їх навантаження. Якщо для паралельно розв’язуваних завдань призначається такий самий ресурс, навантаження на нього може перевищити максимально допустиме. Для вирівнювання навантаження встановіть додаткові зв’язки між завданнями так, щоб завдання, які використовують такий самий ресурс, розв’язувалися послідовно.

Порядок виконання самостійної роботи

1. Ознайомтеся з технічним завданням.

2. Виберіть модель життєвого циклу процесу розробки та впровадження програмного забезпечення, яка, на вашу думку, найбільше відповідає ситуації, що розглядається.

3. Виділіть основні етапи робіт.

4. Виділіть основні завдання всередині окремих етапів робіт.

5. Визначте залежності між завданнями.

6. Визначте порядок розв’язання окремих завдань.

7. Призначте виконавців на оцінювання потреби.

8. Збалансуйте навантаження виконавців.

Зміст звіту

1. Титульна сторінка.

2. Опис виконання роботи відповідно до пп. 1–8.

Питання для самоперевірки

1. Які існують сучасні методології розробки ПЗ?

2. Що передбачає управління ризиками в інжиніринговій компанії?

3. Які етапи має розробка програми управління ризиками в ІТ?

4. Перелічити методи тестування під час набору персоналу.

5. Які існують графічні середовища подання проектної діяльності?

6. Що передбачає управління вартістю проекту?

7. Коротко описати основи розробки бізнес-планів.

Література: [1, с. 54–75; 2, с. 112–125; 5, с. 1–34].

3 КРИТЕРІЇ ОЦІНЮВАННЯ ЯКОСТІ ВИКОНАННЯ САМОСТІЙНОЇ РОБОТИ

У 7-му семестрі студенти виконують самостійну роботу, що складається з 8 завдань. Загальна кількість балів, яку отримують студенти за виконання самостійної роботи, складає максимально 5 балів.

Шкала оцінювання: національна та ECTS

<table>
<tr><td rowspan="2">Сума балів за всі види навчальної діяльності</td><td rowspan="2">Оцінка ECTS</td><td colspan="2">Оцінка за національною шкалою</td></tr>
<tr><td>Для іспиту, курсового проекту (роботи), практики</td><td>Для заліку</td></tr>
<tr><td>90–100</td><td>A</td><td>Відмінно</td><td rowspan="5">Зараховано</td></tr>
<tr><td>82–89</td><td>B</td><td rowspan="2">Добре</td></tr>
<tr><td>74–81</td><td>C</td></tr>
<tr><td>64–73</td><td>D</td><td rowspan="2">Задовільно</td></tr>
<tr><td>60–63</td><td>E</td></tr>
<tr><td>35–59</td><td>FX</td><td>Незадовільно з можливістю повторного складання</td><td>Не зараховано з можливістю повторного складання</td></tr>
<tr><td>0–34</td><td>F</td><td>Незадовільно з обов’язковим повторним вивченням навчальної дисципліни</td><td>Не зараховано з обов’язковим повторним вивченням навчальної дисципліни</td></tr>
</table>

СПИСОК ЛІТЕРАТУРИ

1. Орлов С. А. Технологии разработки программного обеспечения: учебник. СПб.: Питер, 2002. 464 с.
2. Фатрелл Р. Т., Шафер Д. Ф., Шафер Л. И. Управление программными проектами. Достижение оптимального качества при минимуме затрат. М.: Вильямс, 2004. 1136 с.
3. Брукс Ф. Мифический человеко-месяц, или как создаются программные системы. М.: Символ Брукс Ф. Плюс, 2006. 304 с.
4. Орлик С. Введение в программную инженерию и управление жизненным циклом программного обеспечения. – [Электронный ресурс]. – Режим доступа: http://sorlik.ru/swebok-ru/.
5. Мытник С. А. Технологии программирования: учеб. пособие. Томск: ТУСУР, 2007. 196 с.
6. Липаев В. В. Программная инженерия: методологические основы: учебник для вузов. М.: ТЕИС, 2006. 605 с.
7. Попов Ю. И., Яковенко О. В. Управление проектами: учеб. пособие. М.: Инфра–М, 2007. 207 с.
8. Ехлаков Ю. П. Введение в программную инженерию: учеб. пособие. Томск: Эль Контент, 2011. 148 с. [Электронный ресурс]. – Режим доступа: http://edu.tusur.ru/training/publications/141.
9. Бабенко Л. П., Лаврищева Е. М. Основы программной инженерии: учебник. К.: Знання, 2008. 269 с.

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Інженерія програмного забезпечення» (частина І) для студентів денної форми навчання зі спеціальності 123 – «Комп'ютерна інженерія»

Укладач к. т. н., доц. О. Г. Славко

Відповідальний за випуск зав. кафедри КІС М. І. Гученко

Підп. до др. _____. Формат 60×84 1/16. Папір тип. Друк ризографія.
Ум. друк. арк. ____. Наклад _____ прим. Зам. №_____. Безкоштовно.

Редакційно-видавничий відділ
Кременчуцького національного університету
імені Михайла Остроградського
вул. Першотравнева, 20, м. Кременчук, 39600