

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРЕМЕНЧУЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ МИХАЙЛА ОСТРОГРАДСЬКОГО



МЕТОДИЧНІ ВКАЗІВКИ
ЩОДО ВИКОНАННЯ САМОСТІЙНОЇ РОБОТИ
З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
«ТЕХНОЛОГІЇ КРИТИЧНОЇ ПРОГРАМНОЇ ІНЖЕНЕРІЇ»
ДЛЯ СТУДЕНТІВ ДЕННОЇ ФОРМИ НАВЧАННЯ
ЗІ СПЕЦІАЛЬНОСТІ 123 – «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ»

КРЕМЕНЧУК 2018

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Технології критичної програмної інженерії» 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 № 11 від 9 липня 2018 р.

Голова методичної ради _____ проф. В. В. Костін

ЗМІСТ

Вступ.....	4
1 Теми та погодинний розклад лекцій і самостійної роботи.....	6
2 Завдання на самостійну роботу.....	7
3 Критерії оцінювання якості виконання самостійних робіт студентами.....	11
Список літератури.....	12

ВСТУП

Навчальна дисципліна «Технології критичної програмної інженерії» 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 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.

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 проєктованих 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.
1	2	3	4	5	6
Модуль 1					
Змістовий модуль 1 Основні поняття програмної інженерії					
Тема 1 Базові поняття програмної інженерії	16	2	-	-	12
Тема 2 Життєвий цикл програмного забезпечення	16	2	-	6	12
Тема 3 Види життєвого циклу програмного забезпечення	18	4	-	-	12
Усього за змістовим модулем 1	50	8	-	6	36
Змістовий модуль 2 Основні інструменти, методи і технології розробки програмних проектів і систем критичного значення					
Тема 4 Інструменти і методи програмної інженерії	22	4	-	6	12
Тема 5 Технології програмування	20	2	-	6	12
Тема 6 Розробка критичних систем	24	4	-	6	14
Усього за змістовим модулем 2	64	10	-	18	38
ІНДЗ (КР, РГ, к/р)	2	-	-	-	2
Семестровий контроль (іспит)	2	-	-	-	2
Усього годин	120	18	-	24	78

2 ЗАВДАННЯ НА САМОСТІЙНУ РОБОТУ

Тема. Планування робіт по виконанню 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 підзавдання визначаючи послідовність дій.

Для кожного елементарного завдання мають бути визначені:

- ресурси, необхідні для виконання 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 виконуватися паралельно, інші – у суворій послідовності (для виконання одних завдань можуть вимагатися результати виконання інших).

Після визначення залежностей можна переходити до розгляду визначення елементарних завдань за часом. При цьому особливо слід зосередитись на завданнях, які виконуються паралельно.

Якщо план передбачає використання ресурсу 1 для виконання завдань А і Б, то ці завдання не можуть виконуватися паралельно, навіть якщо між ними немає концептуальної залежності. Окрім того, керівник проекту повинен знати завдання, що лежать на критичному шляху. Для того щоб увесь проект був виконаний у строк, необхідно виконувати в 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 повинна 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, СОСОМО II), або може застосовуватися прямий метод оцінювання.

Наступним етапом є визначення зв'язків між завданнями. У більшості засобів планування (наприклад, в 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 виконання завдання Б використовуються результати, напівочікувані під час вирішення завдання А).

Зв'язок типу *початок–початок* означає, що завдання В не може розпочатися до тих пір, поки не почалося виконання 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 виконуватиме і яке обладнання слід використовувати.

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

Після розподілу ресурсів необхідно виконати вирівнювання їх навантаження. Якщо на паралельно виконуваних завданнях призначається один і той же ресурс, навантаження на нього може перевищити максимально допустиме. Для вирівнювання навантаження встановіть додаткові зв'язки між завданнями так, щоб завдання, які використовують один і той ж ресурс, виконувалися 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 виконання 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 СТУДЕНТАМИ

У 10-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

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою	
		Для екзамену, курсового проекту (роботи), практики	Для заліку
90–100	A	Відмінно	Зараховано
82–89	B	Добре	
74–81	C		
64–73	D	Задовільно	
60–63	E		
35–59	FX	Незадовільно з можливістю повторного складання	Не зараховано з можливістю повторного складання
0–34	F	Незадовільно з обов'язковим повторним вивченням навчальної дисципліни	Не зараховано з обов'язковим повторним вивченням навчальної дисципліни

СПИСОК ЛИТЕРАТУРИ

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

Методичні вказівки щодо виконання самостійної роботи з навчальної дисципліни «Технології критичної програмної інженерії» 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 А. В. Луговой

Підп. до др. _____. Формат 60×84 1/16. Папір тип. Друк ризографія.

Ум. друк. арк. _____. Наклад _____ прим. Зам. № _____. Безкоштовно.

Видавничий відділ
Кременчуцького національного університету
імені Михайла Остроградського
вул. Першотравнева, 20, м. Кременчук, 39600