ДОДАТОК А
Слайди презентації

Харківський національний університет радіоелектроніки

“Аналіз різних підходів та інструментів для безперервної інтеграції та розгортання коду”

Виконав: Кварацхелія Т.Е.
Керівник: Костромицький А.І.

Харків, 2019 р.

Харківський національний університет радіоелектроніки

“Аналіз різних підходів та інструментів для безперервної інтеграції та розгортання коду”

Виконав: Кварацхелія Т.Е.
Керівник: Костромицький А.І.

Харків, 2019 р.

ЗАВДАННЯ ДИПЛОМНОЇ РОБОТИ

- Етапи та методолії розробки програмного забезпечення
- Безперервна інтеграція, розгортання та тестування
- Аналіз існуючих інструментів побудови CI/CD/CT
- Проектування комерційного проекту на основі обраного CI/CD/CT інструменту

ЕТАПИ СУЧАСНОЇ РОЗРОБКИ ПЗ

- Стратегія
- Аналіз
- Проектування
- Реалізація
- Тестування
- Впровадження
- Експлуатація та технічна підтримка

МЕТОДОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Методологія розробки ПЗ - це система, яка визначає порядок виконання завдань, методи оцінки та контролю. Моделі розробки ПЗ вибирають, виходячи з напрямку проекту, його бюджету, термінів реалізації кінцевого продукту, а також увагу варто звернути й на характер і темперамент керівника проекту і його команди. Підходи розробки ПЗ відрізняються один від одного тим, як етапи життєвого циклу програмного забезпечення взаємопов'язані між собою всередині циклу розробки

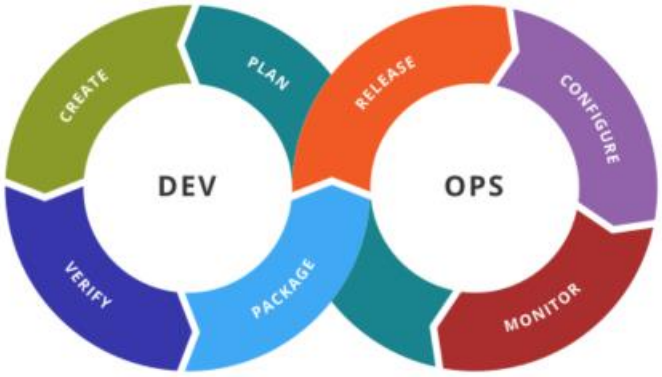
НАЙПОШИРИНІШІ МЕТОДОЛГІЇ

- Waterfall (водоспад)
- V-Model (Крок за кроком)
- RUP (швидка розробка)
- Spiral Model (спіральна модель)
- Extreme Programming (екстремальне програмування)
- Agile Methodology (гнучка методологія)
- DevOps (development and operations)

Agile Methodology

У Agile моделях всі етапи живого циклу виконуються в ході однієї ітерації та готові до впровадження будь-яких змін. Тобто ваш проект ділиться на спринти – відрізки часу, що має принести якийсь результат, як правило один спринт займає від однієї до чотирьох неділь. У кожному спринті є свій власний список завдань, який повинен бути виконаний на конкретній ітерації, і кожна із задач повинна мати свій рівень оцінки. Задачі та прогрес їх виконання обговорюються щоденними зустрічами, в яких команда обтягує, хто що зробив, що збирається зробити і які є важливі проблеми. Окрім цього, на початку спринту проводиться зустріч за плануванням завдання на ітерацію, а в кінці – ретроспективна (зустріч для обговорення результатів).

DevOps Methodology



DevOps (від англ. Development and Operations) – це потомство гнучкої (agile) методології розробки програмного забезпечення – народжене від необхідності йти в ногу з підвищеною швидкістю програмного забезпечення та прохідними методами.

CONTINUOUS INTEGRATION/DELIVERY/DEPLOYMENT/TESTING CI/CD/CT

Безперервна інтеграція (CI), доставка(CD), розгортання (CD) та тестування (CT) – це набір практик розробки програмного забезпечення, який зустрічається у Agile та DevOps метологіях, при якій розробники регулярно об'єднують зміни програмного коду в центральному репозиторії, після чого автоматично виконуються зрібка, тестування та розгортання коду.

Аналіз існуючих інструментів побудови CI/CD/CT

Головна мета виконання цієї роботи полягає у порівняльному аналізі існуючих інструментів побудови CI/CD/CT та проектування працюючого комерційного проету на основі обраного за допомогою аналізу інструмента.

Для того щоб зробити порівняльний аналіз, на основі власного досвіду було обрано декілька найбіль важливих для користувача критеріїв оцінки програмного засобу інструменту, а саме: можливість хостингу, зручність використання, інтеграція та підтримка прогрмних забезпечень, підтримка контейнеризації, бібліотека коду багаторазового використання та ціна.

Порівняльний аналіз інструментів

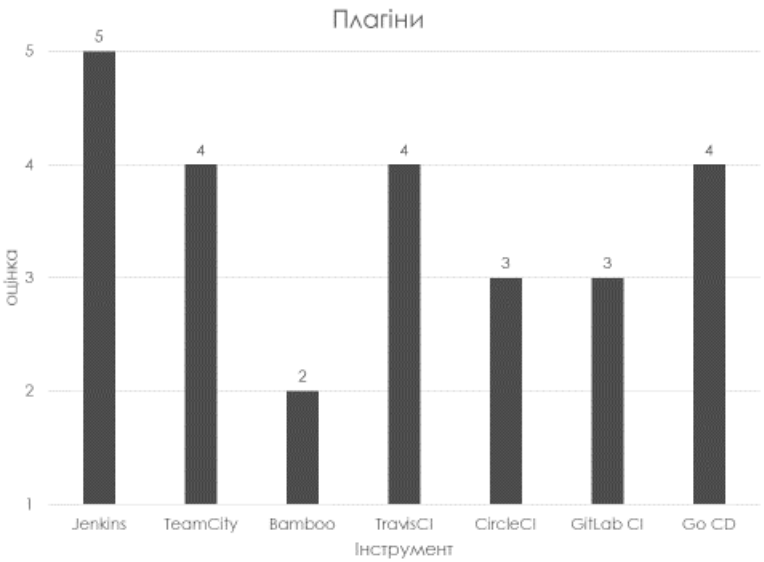
Аналіз оцінок за критеріями виконувався на основі офіційної документації та відгуків і опитувань спеціалістів, які мали досвід з налаштувань CI/CD/CT у комерційних масштабах.

Основним результатом аналізу є нижче наведена порівняльна таблиця

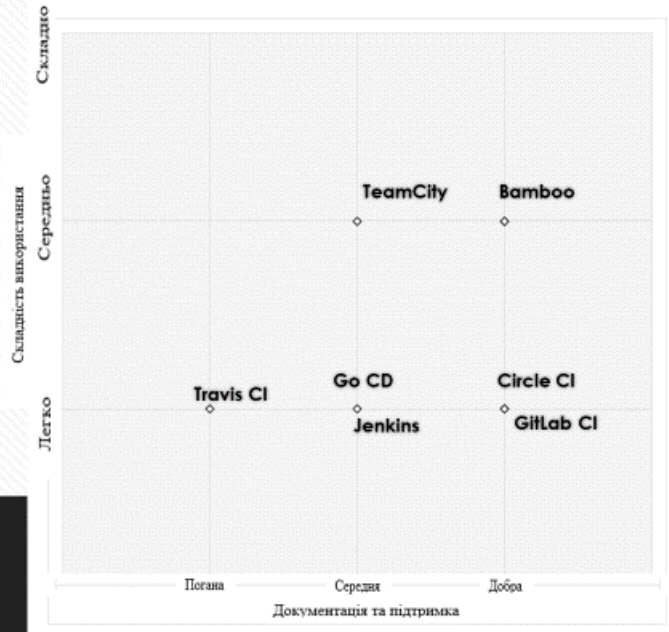
	Jenkins	TeamCity	Bamboo	Travis CI	Circle CI	GitLab	Go CD
Ціна	Безкоштовно	$299-1999	$10-800	$69-489	$50-3150	$0-99	Безкоштовно
Операційна система	Windows, Unix-like, macOS	Windows, Linux, macOS, Solaris, FreeBSD	Windows, Linux, MacOS, Solaris	Linux, MacOS	Linux, IOS, Andorid	Only Unix-like OS	Windows, Linux, MacOS
Хостинг	On Premise*/cloud	On premise	On premise/BitBucket Cloud	On premise/Cloud	Cloud	On Premise/Cloud	On premise/Cloud
Підтримка контейнеризації	+	+	+	+	+	+	+
Плагіни	5	4	2	4	3	3	4
Документація та підтримка	Середня	Середньо	Добра	Погана	Добра	Добра	Середня
Складність використання	Легко	Середньо	Середньо	Легко	Легко	Легко	Легко
Ціль використання	Універсальна система	Для комерційних проектів	Для інтеграції з іншими Atlassain продуктами	Для невеликих проектів та стартапів	Для великих бюджетів та швидких проектів	Для невеликих проектів з визначеною метою	Універсальна система

Оцінка критерію «плагіни» виставлялась за п'ятибальною шкалою, виходячи з кількості існуючих плагінів, їх корисності та відгуків спільноти. Як підсумок щодо можливості використання плагінів, є один лідер – Jenkins. Team City, Bamboo та Go CD також мають добрі показники за цим критерієм.

Порівняльна діаграма якості та підтримки використання CI інструментами додаткових плагінів



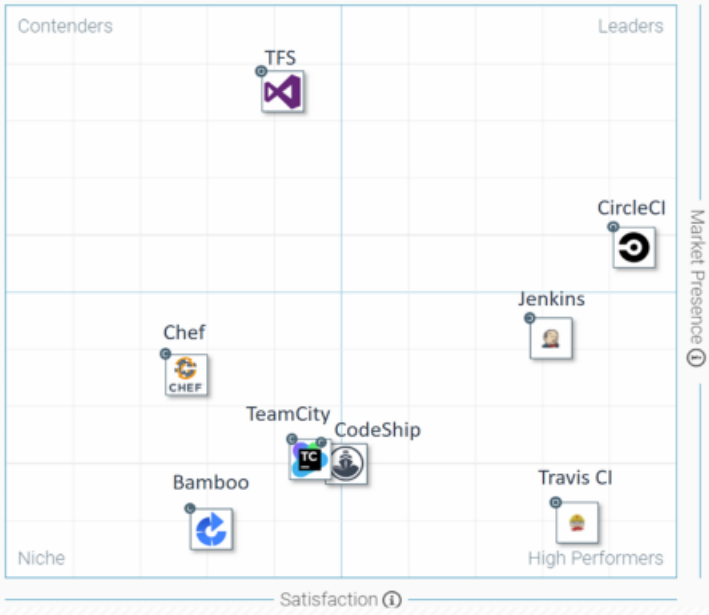
Ця діаграма дуже наглядно відображає такий параметр як складність використання та наявність добре проробленої документації і підтримки з боку користувачів інструменту. За ціма двома критеріями лідирує GitLab CI та Circle CI з дуже добре структурованою документацією та невеликим порогом входу



Графік на основі аналізу документації та складності використання інструменту CI/CD/CT

В якості підсумку огляду загальнодоступних публікацій можна навести наступну діаграму відомого видання G2.

Проаналізувавши результати досліджень видання G2, можна зрозуміти, що деякі розглянуті інструменти є дуже вузько направленими або нішевими, у тай час як безумовними лідерами є Circle CI та Jenkins. Travis CI за даними G2 виявився найбільш продуктивним.

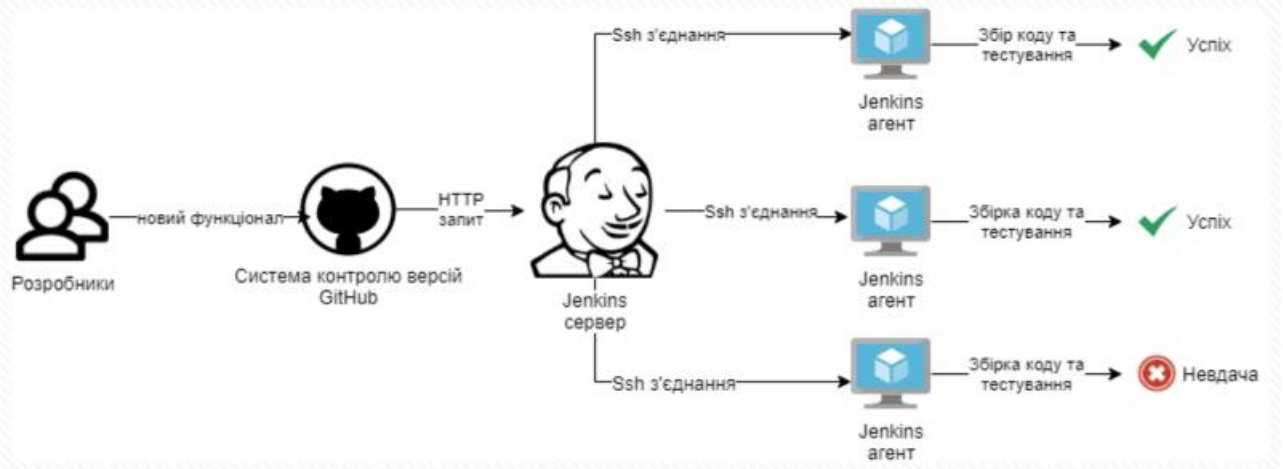


Графік розділення CI/CD/CT інструментів на лідерів, високо продуктивних, нішевих та набираючих популярність

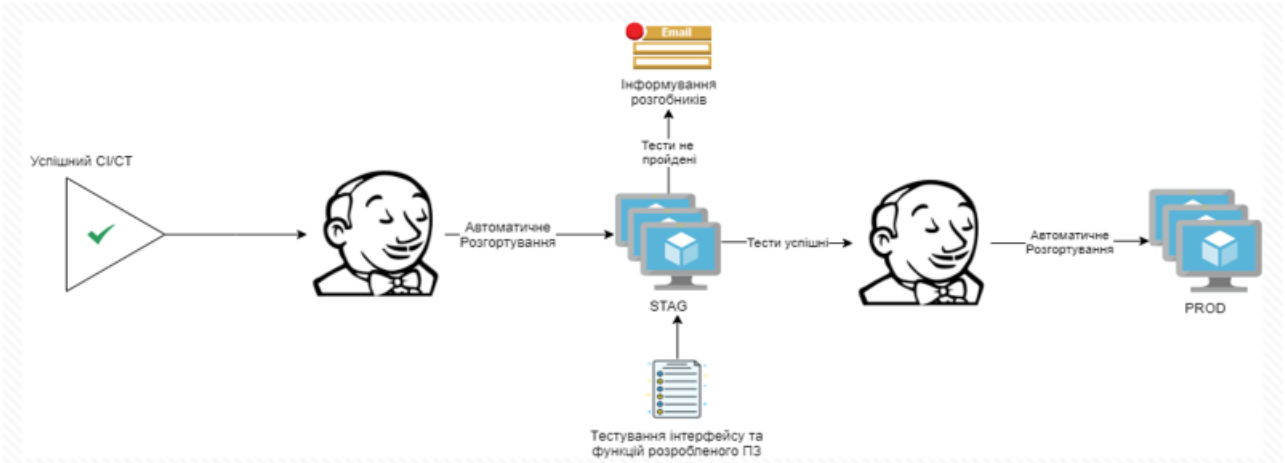
Проектування комерційного проекту на основі обраного CI/CD/CT інструменту

У якості практичного завдання у рамках роботи було спроектовано та налаштовано CI/CD/CT систему комерційного проекту з використанням обраного на основі порівняльного аналізу інструменту.

Завдання: побудувати автоматизований CI/CD/CT для інснуючого проекту з десяти розробників. Мова програмування, яку використовують розробники – Java. Вихідний код зберігається у системі контролю версій Git, у приватному репозиторії GitHub. Команда працює за методом Scrum Кожні два тижні клієнт бажає бачити нову версію працюючого додатку з мінімум непрацюючого функціоналу, та на 99% покритого автоматизованим тестуванням. Деталі задачі описані у пояснювальній записці.



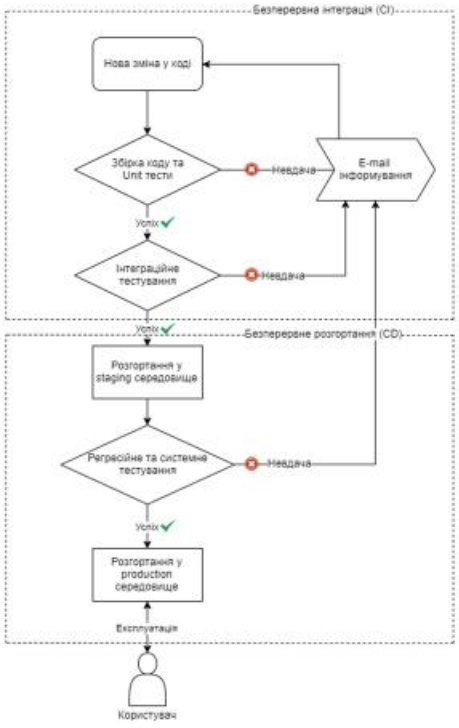
ЗАГАЛЬНА СХЕМА ПОБУДОВАНОГО CI/CT ПРОЦЕСУ З ВИКОРИСТАННЯМ JENKINS



ЗАГАЛЬНА СХЕМА ПОБУДОВАНОГО CD ПРОЦЕСУ З ВИКОРИСТАННЯМ JENKINS

Фінальним етапом виконання проектування було об'єднання CI/CT та CD у єдину систему, яка має наступний алгоритм виконання, наведений на слайді у виді блок-схеми.

Алгоритм роботи налаштованого CI/CD/CT у рамках ТЗ



ВИСНОВКИ

В ході виконання роботи було розглянуто теоретичну частину щодо понять continuous integration, testing, delivery та deployment, їхньої появи на ринку програмного забезпечення. У рамках практичної частини – проаналізовано та порівняно значну кількість інструментів реалізації CI/CD/CT та спроектовано працюючу систему на базі програмного засобу Jenkins.

В результаті отримано значне прискорення розробки, за рахунок автоматизації та зменшенню людської помилки.

ДЯКУЮ ЗА УВАГУ

Позначення	*Найменування*	Дод. відомості
	Текстові документи	
1. ГЮИК.ХХХХХХ.005 ПЗ	Пояснювальна записка	99 с.
	Графічні документи	
	Слайди презентації	20

Змін.	Арк.	Номер докум.	Підпис	Дата	ГЮИК.ХХХХХХ. 005 Д1		
Розроб.		Кварацхелія			Аналіз різних підходів та інструментів для безперервної інтеграції та розгортання коду.	Літ.	Аркуш 1 / Аркушів 1
Перевір.		Костромицький				У	
						ХНУРЕ кафедра ІМІ	
Н. контр.		Золотарьов					
Затв.		Безрук					