## Практична робота №6-6. Порівняння методологій розробки програмного забезпечення. DevOps

**Мета роботи**: засвоєння навичок вибору методології на розробку програмного забезпечення

**Завдання:**

1. Проаналізувати різні методології розробки програмного забезпечення.
2. За результатами аналізу складіть порівняльну таблицю для двох різних за типом методологій.
3. В таблиці для кожної методології повинні бути визначені позиції: суть, дата створення, розробники, принципи застосування, плюси, мінуси, компанії-практики, за яких умов підійде вам, з яких причин не підійде.

По закінченню лабораторну роботу потрібно здати на перевірку викладачеві, надіславши електронною поштою на адресу [**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com). Якщо викладач знаходить помилки чи неточності, він може повернути роботу на доопрацювання.

Файл з роботою повинен мати назву в такому форматі:

**KPZ <Номер групи><Номер лекції / лабораторної> [літера позначення типу роботи L – лекція, R – лабораторна]<Прізвище англійською>**.. Наприклад, **KPZ4101R**buts.doc.

Не копіюйте фрагментів з різних інформаційних джерел, подумайте і викладіть свою точку зору. При наявності робіт-"близнюків" відповідь буде зараховуватися першому за часом надсилання.

Тему в заголовку листа записати

**KPZ<Номер групи>-ЛР<Номер лабораторної>-<Прізвище англійською>**

**Строк виконання цієї роботи ІПЗ-41 – 04.06.2023**

Всі запитання, що виникнуть, надсилайте на електронну адресу викладача, Тему в заголовку листа записати

**KPZ <Номер групи>-Запитання-<Прізвище >**.

**КОНТРОЛЬНІ ПИТАННЯ:**

1. За якими принципами можна класифікувати методології розробки ПЗ?

2. Чому каскадна модель відноситься до прогнозованих методологій?

3. Охарактеризуйте методологію RUP?

4. Які основні ідеї і принципи "гнучких" методологій?

5. Де краще застосовувати "гнучкі", а де прогнозовані методології?

6. В чому сутність методології RAD?

7. Визначте основні принципи екстремального програмування?

8. Що таке програмування «знизу вгору»?

ТЕОРЕТИЧНА ЧАСТИНА

## Порівняльна таблиця методологій

|  |  |  |
| --- | --- | --- |
|  | **Agile** | **Waterfall** |
| **Суть** | Гнучка модель розробки, заснована на ітеративних принципах | Каскадна система розробки, заснована на жорсткій послідовності процесу розробки |
| **Дата створення** | 2001 г. | 1956 року, 1961 р ,1970 р. |
| **Розробники** | Група IT-фахівців (США) | Г. Беннінгтон, Хозьер, В. Уолкер Ройс |
| **Принципи застосування** | * найвищий пріоритет в задоволенні потреб замовника * протягом всього проекту команда і замовник щодня взаємодіють між собою і один з одним * робочий продукт — головний показник прогресу * роботу можна довірити тільки самоорганізованій, мотивованій команді * оптимальні терміни випуску робочого продукту — від 2 тижнів до 2 місяців. | * жорстка послідовність етапів розробки * перехід до нового етапу — тільки після успішного завершення попереднього * фіксована вартість продукту * замовник не залучається до безпосереднього процесу розробки * зміни можуть бути внесені тільки після завершення всього процесу розробки. |
| **Плюси** | 1. високий рівень взаємодії між членами команди проекту 2. швидкий результат (робочий код) в результаті «спринтів» 3. стимулювання змін і поліпшень продукту під час його розробки 4. безпосереднє залучення замовника до робочого процесу. | 1. зрозуміла і чітка схема робочого процесу 2. можливість підрахунку точної кількості витрачених на проект ресурсів 3. не вимагає витрат по налагодженню комунікацій між усіма членами команди. |
| **Мінуси** | * ризик нескінченних змін продукту * велика залежність від рівня кваліфікації та досвіду команди * практично неможливо точно підрахувати підсумкову вартість проекту. | * пріоритет формального підходу до послідовності процесу роботи * неможливість внесення змін замовником до закінчення розробки продукту * в разі нестачі ресурсів страждає якість проекту через скорочення етапу тестування. |
| **Компанії-практики** | Unilever, ряд банків (Альфа Банк, Home Credit, Райффайзен Банк і т.д.) | Cisco Ericsson AB, Toyota (до 2010) |
| **Підійде вам, за умов** | 1. над проектом працює досвідчена, висококваліфікована команда 2. ви працюєте над стартапом 3. потрібно швидко отримати робочу версію продукту 4. замовник виступає в якості партнера, а не інвестора 5. продукт розробляється в сфері, схильній до постійних змін. | 1. більша частина або вся робота над проектом проводиться на аутсорсі 2. у вас є чітка концепція продукту, який хочете отримати 3. ви не обмежені в часі і ресурсах створення продукту 4. створення продукту або бізнесу побудовано на дотриманні суворої послідовності виконання задач. |
| **Не підійде з причин** | * ви не готові витрачати додаткові ресурси на налагодження щоденної стабільної комунікації між усіма учасниками процесу * продукт повинен бути створений до конкретного терміну * бюджет проекту строго обмежений * вам потрібна детальна документація по всім процесам розробки. | * ви хочете створити інноваційний продукт або великий проект * ви не впевнені в концепції пропонованого проекту * фінансові ресурси не є ключовим обмежувачем в вашому проекті. |

Таблиця створена за матеріалами статті "Agile чи Waterfall — який варіант відповідає вашому бізнесу?" - URL: https://worksection.com/ua/blog/waterfall-vs-agile.html

## DevOps

**DevOps** (Development and Operations) –  це методика, спрямована на забезпечення ефективної взаємодії розробників та кінцевих користувачів продукту та оптимізацію всіх процесів життєвого циклу ПЗ.

**DevOps – це набір практик,** що поєднує **розробку програмного забезпечення** з його обслуговуванням і експлуатацією. Його назва відображає ці дві частини: **Development and Operations** (розробка та операції). DevOps виник із колекції попередніх практик. Серед них – система **Agile development, Toyota Way та Lean manufacturing.** Термін DevOps став широко відомий на початку 2010-х років.

**Основна мета DevOps – скоротити час** між тим, коли **внесли зміни у код** і тим, коли **клієнт ці зміни відчув,** не впливаючи при цьому на надійність продукту. Фахівець прагне узгодити цілі розвитку з потребами організації для створення цінності бізнесу.

**Інженер DevOps – це** ІТ-спеціаліст **широкого профілю,** який повинен мати грунтовні знання як у розробці, так і в різних операціях, включаючи **кодування, управління інфраструктурою, системне адміністрування та ланцюжки інструментів DevOps**. Інженери DevOps також повинні володіти навичками міжособистісного спілкування, оскільки вони працюють з різними командами компанії, щоб створити середовище для більш ефективної співпраці.

DevOps -інженери повинні добре **розуміти загальну архітектуру системи,** ініціалізацію та адміністрування. Також DevOps-інженер повинен мати досвід роботи з **традиційним набором інструментів** розробника та практиками, такими як:

* використовувати систему **контролю первісного коду (Sourse Code Control System);**
* надавати та отримувати огляду коду **(Code Review);**
* писати **модульні тести;**
* бути знайомим з **принципами Аgile.**

**Існує думка, що** у цій професії **не потрібно технічних знань**!

Роль інженера DevOps буде **відрізнятися від компанії до компанії,** але незмінно передбачає, що фахівець буде **поєднувати розробку релізів, забезпечувати інфраструктуру та управляти нею; бути системним адміністратором, забезпечувати безпеку всього, що є на проєкті, та просувати філософію DevOps.**

**DevOps Engineer** – це спеціаліст, який **бере участь у всіх етапах життєвого циклу продукту.** Цей фахівець забезпечує тісну **кооперацію між** командами розробників**,** щоб  оптимізувати їхні робочі процеси і, відповідно, **допомагає скоротити час, за який програмний продукт потрапить до кінцевого користувача.**

Коли ми говоримо про DevOps-інженера потрібно розуміти, що **цей фахівець не є якимось конкретним вузькопрофільним спеціалістом.** Ці фахівці приходять на посаду з різних професій. Наприклад, **інженер DevOps може почати як розробник програмного забезпечення, який контролює аспекти ІТ-операцій.** З іншого боку, інженер DevOps може бути в минулому системним адміністратором, оскільки він отримав знання про кодування, створення сценаріїв, інтеграцію та тестування. Але саме до обов’язків інженера DevOps входить зміна бізнес-процесів таким чином, щоб вирішити організаційні проблеми і покращати бізнес-результати.

Незважаючи на широкий і різноманітний діапазон ролей DevOps, **є деякі загальні навички та вміння, на які вам потрібно звернути увагу, якщо ви націлені стати "девопсом", то ви повинні мати:**

* досвід адміністрування ОС, таких як **Linux і Windows;**
* значний досвід роботи з рядом **інструментів автоматизації** та керування конфігураціями, такими як традиційні сценарії, а також більш специфічними інструментами**;**
* чітке розуміння програмування та створення сценаріїв поширеними **мовами програмування, такими як PHP, Python, Perl і Ruby;**
* знання принаймні однієї **основної мови кодування, такої як C++ або Java;**
* здатність ідентифікувати, оцінювати та **інтегрувати різні технології** з відкритим кодом , а також хмарні сервіси;
* серйозний досвід ІТ-апаратного забезпечення та роботи з практичним досвідом **встановлення серверів,** сховищ і мережевих пристроїв, ініціалізації та моніторингу. Це зазвичай доповнюється знаннями найкращих ІТ-практик для відмовостійких операцій з високою доступністю;
* здібності до **підтримки команд та співпраці з ними,** а також підтверджені навички взаємодії з клієнтами;
* грунтовне знання **технологій віртуалізації**, таких як **VMware vSphere** для віртуальних машин, а також досвід роботи з контейнерними технологіями, такими як **Docker і Kubernetes;**
* перевірений **досвід роботи з інструментами CI/CD, де** CI (Continuous Integration) — це неперервна інтеграція, а CD (Continuous Delivery) — неперервна доставка. Цей набір методик дозволяє розробникам частіше і надійніше розгортати зміни в програмному забезпеченні.**). До таких інструментів відносяться Microsoft GitHub, Atlassian Jira та Confluence, Red Hat Ansible, Prometheus та Jenkins;**
* грунтовний практичний досвід роботи з **публічними хмарними ресурсами** та службами, такими як **AWS, Microsoft Azure і Google Cloud;**
* досвід роботи з різноманітними інструментами ІТ-моніторингу та управління, такими як **Cloudflare і Datadog;**
* знання про те, як усунути несправності та вирішити технічні проблеми в тестовому та робочому середовищах.

## Роль DevOps-інженера та його/її обов’язки

На сьогодні DevOps інженер – одна з найнеобхідніших позицій в компаніях, де розробляються продукти. Чому так? Щоб відповісти на це питання, для початку нам необхідно розібратися із самою методологією і DevOps інженерами.

## Що таке DevOps?

DevOps або ж Development Operations – це не просто професія, а певна методика для розв’язання проблем у взаємодії між командами експлуатації та розробки. До виникнення DevOps була наступна ситуація: розробники запускали код в експлуатацію, при виникненні будь-яких проблем у команди експлуатації, відповідальність перекладалася на першу команду. Це призводило до того, що багато продуктів випускалися з запізненням або ж з вадами. Також на кінцевий продукт впливало і велика кількість змін, яких важко було відстежити в процесі супроводження.

Методологія DevOps була створена для розв’язання подібних проблем. Іншими словами, DevOps – це міст між командами розробки та експлуатації.

## Тоді хто такі DevOps інженери?

DevOps інженери – це фахівці, які розбираються з усіма можливими процесами циклу розробки й експлуатації. До них відносяться: розробка, тестування, архітектура продукту, оцінка ризиків захисту, різні способи автоматизації, підтримка продукту як до, так і після випуску. Більш того, фахівець розуміє, як працюють обидві команди розробників і експлуататорів.

DevOps умовно ділиться на кілька субролей:

* Build Engineer – професіонал, який відповідає не тільки за збірку коду, але і за розбір конфліктів і підтягування різних залежностей;
* Release Engineer – людина, яка відповідає за доставлення коду від команди розробників в експлуатацію, за фінальну гілку для тестування та ін.
* Automation Engineer – фахівець, який працює над автоматизацією всіх продуктів.
* Security Engineer – людина, яка відповідає за коректність тестів з безпеки, перебування слабких місць в компонентах продукту.

На ділі ж всі ці субролі виконує одна людина або ж кілька ролей інший фахівець. Буває так, що розробник може виконувати завдання Build Engineer. А з автоматизацією працюють системні адміністратори. В результаті DevOps інженер тільки автоматизує та робить остаточне опрацювання та автоматизація виробництва та доставки коду в експлуатацію.