1.09.2023

МДК 04.02 Обеспечение качества функционирования компьютерных систем (DevOps)

Каскадные модели разработки

**Waterfall**

* Каждый этап один за другим – нельзя вернуться назад
* Этапы циркулируют только после завершения последнего этапа



Гибкие модели разработки

**SCRUM**

* В конце спринта имеется готовый продукт
* Гибкое изменение требований
* Быстрое реагирование на изменение трендов
* Бюджет не фиксирован



Основные этапы разработки ПО

1. Анализ
2. Разработка
3. Сборка
4. Тестирование
5. Развертывание
6. Релиз (деплой)
7. Сопровождение
8. Мониторинг

**DevOps**

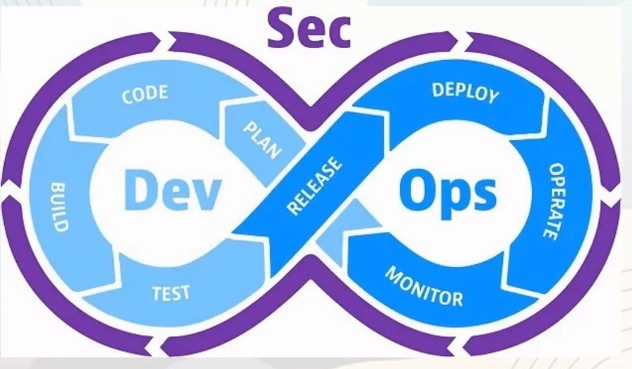
* Бесперебойная поставка ПО с помощью непрерывной интеграции рабочих процессов
* Ускорение этапов разработки
* Снижение ошибок, связанных с человеческим фактором



**DevSecOps**

* - Интегрирует безопасность в процессы разработки и автоматизации

Development + Security+ Operations = DevSecOps



**Концепции составляющие DevOps**

**Инфраструктура как код**

Методология, которая предлагает использовать архитектурный подход к управлению инфраструктурными ресурсами, а именно автоматизировать создание и управление инфраструктурными компонентами с помощью программного кода.

Основная идея заключается в том, что вся инфраструктура, такая как виртуальные машины, сетевые настройки, хранилища данных и другие ресурсы, описывается в виде кода, который может быть управляем исходными файлами, хранящимися в системе контроля версий.

**CI/CD**

Методология разработки программного обеспечения, целью которой является автоматизация и ускорение процесса разработки, тестирования и доставки программного продукта.

Цель СІ/CD состоит в том, чтобы обеспечить быстрое время от

идеи до развертывания новых функций или исправлений ошибок, с учетом меньшей вероятности ошибок и конфликтов, а также поддержания высокого уровня качества кода и процесса разработки.

**Состав**

-Непрерывная интеграция(Continuous Integration)

-Непрерывная доставка(Continuous Delivery)

- Непрерывное развертывание (Continuous Deployment)

Что такое система контроля версий и для чего она нужна

Система контроля версий- это специальное программное обеспечение, которое используется для управления изменениями в файловой системе, отслеживания и контроля версий документов или кода программы. Она позволяет разработчикам и команде проекта работать над файлами вместе, отслеживать изменения, управлять их версиями и восстанавливаться к предыдущим состояниям проекта.

Распределенная VS Централизованная системы контроля версий

Архитектура: Централизованная система контроля версий имеет единую центральную базу данных, где хранятся все файлы и изменения.

В распределенной системе каждый пользователь имеет собственную копию репозитория, включая всю историю изменений.

Работа в офлайн-режиме: Распределенная система позволяет работать в офлайн-режиме, так как каждый пользователь имеет полную копию репозитория. В централизованной системе требуется подключение к центральному серверу для выполнения большинства операций.

Управление правами доступа: В централизованной системе управление правами доступа к репозиторию обычно происходит на уровне центрального сервера. В распределенной системе каждый пользователь может выполнять операции с собственной копией репозитория, что позволяет более гибко управлять правами доступа.

Скорость работы: Распределенная система обычно быстрее выполняет операции, так как все данные хранятся локально у каждого пользователя.

В централизованной системе операции происходят через удаленное соединение, что может быть медленнее при большом количестве пользователей.

snapshot-based

Система контроля версий с механизмом снимков (например,

Git, Mercurial) хранят все версии файлов и директорий в форме полных копий (снимков) состояния проекта в определенный момент времени. Когда вы делаете коммит, система фиксирует все измененные файлы и создает новый снимок, который включает в себя полные копии всех измененных файлов. Это позволяет быстро переключаться между разными версиями проекта и осуществлять обход старых коммитов.

delta-based

Система контроля версий со списком изменений (например,

Subversion) хранят только разницу (delta) между последующими версиями файлов. Когда вы делаете коммит, система сохраняет только изменения, сделанные в файлах, относительно предыдущего коммита, в виде списка изменений. На самом деле, система часто хранит несколько версий файла, чтобы быстрее обрабатывать запросы, основанные на списке изменений. При переключении на предыдущие версии проекта, СКВ применяет все изменения последовательно, чтобы восстановить запрошенную версию проекта.