데브옵스(DevOps)의 이해

개발팀과 IT 운영팀의 작업에 대한 공유 방식으로 조직 내 여러 팀 간 의사소통과 협업을 활성화 시키는 운영 방식으로 보면 되겠다.
개발팀과 IT 운영팀이 나누어져 있는 상황에서, 개발팀은 프로그램의 개발만을 IT 운영팀은 기존 시스템의 운영만을 담당하므로 개발자와 운영자에 대한 구분과 불신이 나타나게 되고 이렇게 되면 개발한 프로그램에 대한 배포와 류를 잡아 프로그램을 안정화하는 과정이 오래 걸려 비효율을 초래하게 되는데
개발자와 IT 운영자를 통합하는 Devops를 적용하여 개발부터 운영까지 한번에 처리해보자는 것이다.

- 장점

1. 개발팀과 운영팀간의 의사소통 증가로 생산성이 증가
2. 한 곳에서 개발부터 검증, 배포까지 전체를 담당하게 되어 개발과 배포 속도가 빨라짐
3. 구성원에게 개발 책임감과 코드의 소유권을 높여줘 개발 프로세스 간소화

- 단점

1. 다양한 팀이 모여 업무 역할이 변경되기 때문에 활성화되기 위해서는 충분한 시간이 필요
2. 코드를 자주 배포할 필요가 없다면 비용만 늘어남
3. 포괄적인 자동화 도구가 필요함

Devops는 도구가 아니라 일을 이렇게 하자는 방법론이지만 주요 구성 내용은 다음과 같다.

- 코드 리포지토리(Code Repositories)
  - 코드 수정 내용 보관 및 타 개발자들이 변경된 코드 적용
  - ex) Git, SVN
- 아티팩트 리포지토리(Artifact Repositories)
  - 프로젝트 수행 시 만든 산출물(설계 문서, .jar 파일 등)을 버전별로 보관
  - ex) JFrog, Nexus Repository
- 컨테이너(Containers)
  - 가상 환경에 접속한 후 해당 환경에 맞추어 세팅해주면 개별 시스템별로 세팅해줄 필요 없이 컨테이너에 맞춘 대로 개별 시스템에 동일하게 세팅이 가능해짐
  - ex)Docker, Microsoft Hyper-V
- CI/CD
  - CI(Continuous Integration) 지속적인 통합
  - CD(Continuous Deploy or Continuous Delivery) 지속적인 배포
  - 품질관리를 위해 지속적인 빌드, 테스트 과정 지원, 지속적인 배포 가능한 기능을 포함
  - ex)Jenkins, Travis CI

Ref

https://wakestand.tistory.com/259

데브옵스, Devops, 밤즌

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:pm:devops:%EB%8D%B0%EB%B8%8C%EC%98%B5%EC%8A%A4_devops_%EC%9D%98_%EC%9D%B4%ED%95%B4&rev=1647837280

Last update: **2022/03/21 13:34**