DevOps

- description : DevOps
- author : 밤즌
- email : bjlee@repia.com
- lastupdate : 2022-03-16

Intro

데브옵스(Devops)란?

- 데브옵스(DevOps)는 개발(Development)과 운영(Operations)의 합성어
- 개발과 운영의 경계를 허물고 하나의 팀으로서 소통, 협업 및 통합을 강조하는 개발 환경이나 문화를 뜻 함.

Case Study

데브옵스(DevOps)의 이해

- 개발팀과 IT 운영팀의 작업에 대한 공유 방식으로 조직 내 여러 팀 간 의사소통과 협업을 활성화시키는 운영 방식
  - 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 내용과 도구의 예시는 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

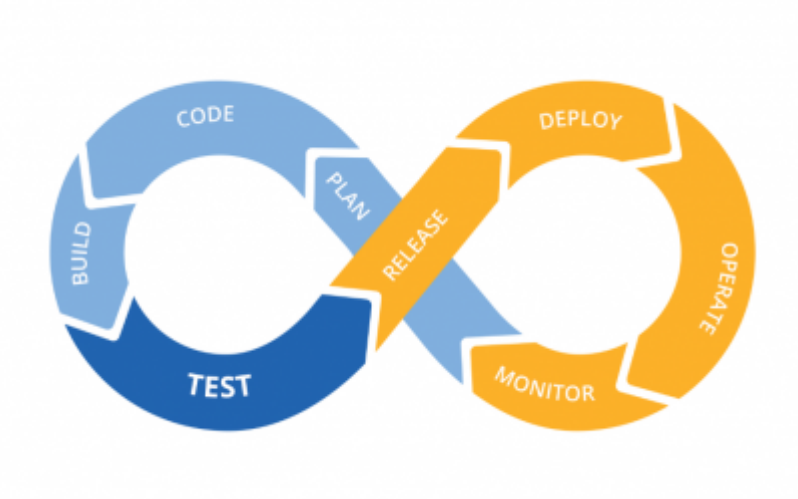
- 컨테이너(Containers)
    - 가상 환경에 접속한 후 해당 환경에 맞추어 세팅해주면 개별 시스템별로 세팅해줄 필요 없이 컨테이너에 맞춘 대로 개별 시스템에 동일하게 세팅이 가능해짐
    - ex)Docker, Microsoft Hyper-V

- 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

데브옵스(DevOps)의 이점

1.속도
Micro Service Architecture(단일 애플리케이션을 작은 서비스의 집합으로 구축하며,
각 프로세스는 자체 프로세스에서 실행되고 잘 정의된 인터페이스를 통해 다른 서비스와 연결되는 설계기법) 및 CD(Continuous Delivery)등을 사용하여
팀에서 서비스를 주도적으로 운영하여 수정된 코드들을 더 빠르게 릴리스 할 수 있다.
2.신속한 제공
CI(Continuous Integrity) 및 CD(Continuous Delivery)등을 통해 빌드에서 배포까지 자동화시켜
릴리스의 빈도와 속도를 개선하여 제품을 더 빠르게 업데이트 할 수 있다.
새로운 기능 및 버그 수정 속도가 빨라 고객의 요구에 더 빠르게 대응할 수 있게 된다.
3.안정성
CI(Continuous Integrity) 및 CD(Continuous Delivery)등을 통해 변경사항을 안전하게 작동하는지 업데이트마다 테스트 해주어
애플리케이션에 안전성 및 인프라 변경의 품질의 보장해준다.
4.확장
규모에 따라 인프라와 개발 프로세스를 운영관리하여 주며 자동화와 일관성의 지원 이점을 살려
복잡한 시스템이나 변화하는 시스템을 효율적으로 관리할 수 있도록 해준다.
또한 코드형 인프라(버전 관리 및 CI(Continuous Integrity)과 같은 코드와 소프트웨어 개발 기술을 사용하여 인프라를 예측하고 관리하는 방식으로
클라우드의 API 중심 모델 사용을 통해 개발자와 시스템 관리자가 수동으로 리소스를 설정 및 구성할 필요 없이
프로그래밍 방식으로 큰 규모로 인프라와 상호 작용하도록 해주는 방식)등을 사용하여
개발, 테스트, 프로덕션 환경을 반복 가능하고 효율적인 방식으로 관리 할 수 있도록 하여준다.

5.협업 강화
개발팀과 운영팀은 서로 긴밀하게 협력하며, 책임을 강조하는 가치를 통하여 효과적인 팀을 구축한다.
6.보안
자동화된 규정 준수 정책 및 세분화된 제어 및 구성 관리 기술을 통하여 제어를 유지하고, 규정을 준수하며 작업을 진행해 나아가는데
보안도 지켜 나아갈 수 있도록 하여준다.



데브옵스(DevOps)와 애자일(Agile)의 유사점

1.둘 다 생산성 향상을 위해 노력한다.
Agile과 DevOps는 서로 다른 작업 접근 방식을 가지고 있지만 비즈니스 생산성 향상이라는 공통 목표를 가지고 있습니다.
2.둘 다 린 철학과 응용 프로그램을 채택한다.
Agile 또는 DevOps 방법론을 따르는 팀은 린 철학을 빠르게 활용하고 구현할 수 있습니다.
불필요한 대기 시간이나 제품 결함 및 기타 초과 프로세스로 인한 추가 비용과 같은 비효율성을 식별하고 제거할 수 있습니다.
3.후속 협업 프로세스
프로젝트와 관련된 구성원은 이러한 소프트웨어 개발 방법론 중 하나를 따르면 훌륭한 협업을 수행할 수 있습니다.
다양한 팀이 개발 단계를 업데이트하고 같은 페이지에 있는 것이 쉬울 것입니다.

데브옵스(DevOps)와 애자일(Agile)의 차이점

1.접근
애자일 방법론을 통해 팀은 점진적 개발 접근 방식으로 작업할 수 있습니다.
여기에는 클라이언트와 개발 팀 간의 지속적인 반복과 커뮤니케이션이 포함됩니다.
DevOps 방법론은 비즈니스 운영을 개선하기 위해 배포 단계에 더 중점을 둡니다.
이 방법을 따를 때 프로젝트 관리자는 전달 프로세스를 위해 두 가지 옵션을 사용할 수 있습니다.
1) 사용 가능한 제품 업데이트를 실시간으로 배포한다.
2) 모든 업데이트가 표시될 때까지 기다렸다가 모두 함께 배포한다.
2.문서화 및 커뮤니케이션
스크럼 플랫폼은 애자일 방법론을 구현하는 데 가장 적합합니다. 처음에는 소프트웨어를 구축하기 위해 발표되었습니다.
그러나 이는 팀이 오늘날 다른 복잡한 작업을 완료하는 데 도움이 됩니다.
민첩한 접근 방식을 통해 팀 구성원은 정기적인 회의를 열고 의사 소통을 개선할 수 있습니다.
결과적으로 프로젝트 관리자 또는 팀 리더는 팀 구성원의 작업을 효율적으로 관리할 수 있습니다.
DevOps 접근 방식은 팀이 소프트웨어 수명 주기의 다양한 단계에 대한 포괄적인 문서로 작업하는 데 도움이 됩니다.
운영 팀은 개발 프로세스에 대한 철저한 지식을 얻고 그에 따라 배포 준비를 할 수 있습니다.
3.작업 과정
애자일 방법을 따르는 팀은 교차 기능과 함께 작동합니다.
각 팀 구성원은 프로젝트의 전반적인 진행 상황에 대한 아이디어를 가지고 할당된 작업에 노력을 기울입니다.
모든 개인이 자신의 역할에 대해 명확하게 이해하고 있기 때문에 보다 빠르고 잘 조직화된 품질 개발 프로세스가 가능해집니다.
반면에 프로젝트 팀이 DevOps 방식을 따르려면 개발 및 운영 부서의 __적극적인 참여__가 필요합니다.
그들은 가능한 최상의 결과를 달성하기 위해 프로세스에서 긴밀한 조정 작업을 수행해야 합니다.

Term

CI,CD

Ref

https://aws.amazon.com/ko/devops/what-is-devops/?nc1=h_lshttps://aws.amazon.com/ko/devops/what-is-devops/?nc1=h_ls

데브옵스, DevOps, 밤즌

☐ 이 페이지 초안 작성 부탁합니다. (다른 페이지를 보거나, 주위에 물어서요~~) Convention과 Consensus가 중요해요) (할당된 사용자 밤즌. 마감일 2022-03-18)

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&rev=1647854994

Last update: **2022/03/21 18:29**