데브옵스 VS 기존의 접근 방식

- description : 데브옵스 VS 기존의 접근 방식
- author : 밤즌
- email : bjlee@repia.com
- lastupdate : 2022-03-21

DevOps 진행	**기존의 접근 방식**
– 협업 중심. 성공적인 DevOps는 신속하고 신뢰성있는 소프트웨어의 개발과 전달을 보장하기 위해 개발팀과 IT 운영팀의 성공적이고 지속적인 협업 능력에 의존합니다.	– 사일로 중심. 기존의 접근 방식은 협업에 대해서는 “ 다짜고짜 떠넘기기 ” 에 의존합니다. IT 운영자는 실제 운영 환경에서 소프트웨어 배포와 관리를 담당하며, 개발팀은 최소한의 지원을 제공합니다.
– 상당한 수준의 (또는 전체적) 구조화와 자동화. DevOps 진행은 개발 환경에서의 작동이 프로덕션 환경에서의 작동을 보장하는 환경 프로비저닝과 구성에서 속도, 일관성, 반복성을 제공하는 자동화에 의존합니다. 또한 구조적 접근 방식은 오류 복구를 반복 가능한 자동화만큼 빠르게 바꿔주므로 롤백과 복구가 편해집니다.	– 눈송이 중심 및 대부분 수동. 기존의 접근 방식은 프로젝트를 제대로 수행하거나 신뢰 가능하게 반복하거나 신속하게 진행하기 어렵습니다. 또한 개발 및 프로덕션 인프라스트럭처와 구성 패리티를 신속하고 일관되게 프로비전하는 능력이 부족하므로 많은 문제에 봉착하는 경향이 있습니다.
– 셀프 서비스 중심. DevOps 중심 조직은 협업 및 자동화 프레임워크를 구축해 개발자와 IT 운영자에게 상대방에게 방해되지 않고 독립적으로 일하도록 자율권을 줍니다. 예를 들어 개발자는 IT 운영자의 직접적인 프로비저닝을 기다리지 않고 신속하게 개발/테스트 환경을 자체적으로 프로비전할 수 있습니다.	– “ 정보 기술(IT) 티켓 ” 중심. 기존의 엔터프라이즈 접근 방식에서 IT 운영자는 쉽게 자동화 가능한 IT 티켓 관리 작업을 수행하고 반복적이고 복잡한 수동 프로비저닝과 구성을 진행해야 합니다. 그 결과 프로비저닝, 배포, 크기 조정, 다른 소프트웨어 전달 및 관리 활동이 상당히 복잡해지고 지연됩니다.
– 비즈니스 중심. DevOps 조직은 공동으로 비즈니스 성공을 추진하는데 집중한다. 그러므로 소프트웨어 전달 성공에 대한 책임도 공동으로 집니다.	– 기능 중심. 기존의 접근 방식은 개발자와 IT 운영자가 그들 기능에 집중하되 전체적인 성공에 대해서는 별다른 책임을 부여하지 않았습니다. 그 결과 여러 에러 상황이 발생해 많은 비난을 받고 조직내에서 마찰이 발생합니다.
– 변경 맞춤형. DevOps 진행은 신속하고 반복 가능하게 자동화하도록 디자인됩니다. 신속한 오류 복구뿐만 아니라 신속한 변경 처리도 수행하게 빌드됩니다. 신속하게 진행하기 위해 빌드되는 것입니다.	– 변경 반대. 기존의 접근 방식에서는 손댔다가 빨리 복구할 수 없게 될까봐 프로덕션 배포를 변경하지 않습니다. 전통적인 접근 방식에서는 변경과 업데이트를 최소화하고 조직에게는 천천히 진행할 것을 간접적으로 권장합니다.

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_vs_%EA%B8%B0%EC%A1%B4%EC%9D%98_%EC%A0%91%EA%B7%BC_%EB%B0%A9%EC%8B%9D&rev=1647913762

Last update: 2022/03/22 10:49