

7 Years as a Developer, Day 1 as a Manager

<https://product.kyobobook.co.kr/detail/S000001810222>

원온원미팅

원하는 것을 생각하는 데 시간을 써라

자신을 스스로 책임져라

주니어에게는 멘토형 매니저이자 시니어에게는 변호사형 매니저 일수도 있다.

알파 킷

- 탁월하고 효율적으로 일하는 개발자
- 탁월함의 문화를 만들려고 두려움의 문화를 만드는 매니저가 되는 경향 존재

멘토가 집중해야하는 3가지 활동

- 호기심과 열린 마음 소유
- 상대방의 언어 듣고 말하기
- 인맥 관리하기

RTR 기술

"테크리드는 소프트웨어 개발 팀을 책임지는 리더로, 적어도 30%정도의 시간을 팀과 코드 작성을 위해 사용한다." - 패트릭 쿠어

테크리드 = 시스템 아키텍트 및 비즈니스 분석가 + 프로젝트 기획자 + 소프트웨어 개발자
및 팀 리더

승자의 돌

애자일 개발 방법론 ~ 건강한 프로세스 리더십

설명의 중요성

작업을 작게 나눈다 : 스프레드시트, 간트 차트 등 -> 일을 어렵게 만드는 세부 사항과 문제가 될 수 있는 부분은 끝까지 신경쓴다. -> 프로젝트 시작하고 진행하며 계획 수정한다. -> 계획 프로세스에서 얻은 통찰로 변경된 요구사항을 관리한다. -> 프로젝트 완려 시점이 가까워지면 세부 사항을 다시 검토한다.

훌륭한 테크리드 되는 방법

- 아키텍처 이해하기
- 팀 플레이어되기
- 기술 결정을 주도하기
- 의사소통

새로운 팀원과 관계 맺기

- 신뢰 관계 구축하기
- 30/60/90일 계획 세우기
- 신규 직원용 문서 갱신하기
- 관리 스타일과 기대 사항 소통하기
- 신규 직원의 피드백 받기

위임과 포기는 다르다.

효율적으로 위임하기 위한 실질적인 조언

- 팀원의 목적을 통해 어떤 세부 사항을 파악해야 할지를 판단하기
- 팀원 만나기 전 시스템에서 정보 수집하기
- 프로젝트 단계에 따라 확인할 부분 달리하기
- 코드 및 시스템 표준을 설정하기
- 독립적인 태도로 정보 개방하기

360도 성과 평가하기

- 충분한 시간 들이고 일찍 시작하기
- 1년의 성과 고려하기
- 구체적인 예를 들고, 동료 평가에서 발췌한 내용 이해하기
- 성취와 감정에 많은 시간 할애하기
- 개선할 부분이 있는 경우에는 그 부분에 집중하기
- 평가에 대해 충분히 논의할 일정 잡기

엔지니어링 리드 : 코드 작성에 상대적으로 적은 시간을 쓰지만, 팀의 작업을 방해하거나 작업 속도를 떨어뜨리지 않는 정도의 버그를 수정하고, 작은 기술 산출물에 관여. 코드 작성 말고도 팀의 성공을 위해 개발 프로세스의 병목 지점과 장애물 찾고 이를 제거하는 책임지는 매니저

좋은 의사 결정 내리는 방법

- 데이터 중심의 팀 문화 만들기
- 제품 역량 강화하기
- 미래 내다보기
- 의사 결정과 프로젝트 결과에 대해 검토하기
- 프로세스와 일상 업무에 대해 회고하기

갈등관리에서 하지 말아야 할 것

- 협의나 투표나 양자택일 자제하기

- 객관적인 결정이 가능한 명확한 프로세스 마련하기
- 금방 터질 이슈 외면하지 않기
- 드라마 만들지 말고 해결하기
- 다른 팀에 화풀이하지 않기
- 친절하게 행동하기

프로젝트 관리에 대한 경험 법칙

- 어느 것도 애자일 프로젝트 관리 방법 대체 불가
- 분기별로 엔지니어 당 10주의 생산적인 엔지니어링 주가 있다
- 일상적으로 필요한 일반적인 유지 보수 업무를 하기 위해 20%정도의 시간을 배정한다
- 마감 시간이 다가오면 아니오라고 말하는 것이 매니저의 책임이다.
- 빠른 추정을 할 때는 두배의 법칙을 사용하지만, 긴 작업을 추정하는 계획 시간을 따로 가져야한다.
- 추정을 통해 팀에 가져올 내용을 선택 할 수 있도록 한다.

엔지니어링 디렉터 : 여러 제품군이나 기술을 어우르며 팀을 리드. 매일 코딩 할 필요는 없으나 조직의 전반적인 기술 역량을 책임지고, 필요한 경우 교육과 채용을 통해 전체 팀의 역량을 이끌고 성장시키기 위해 탄탄한 기술 배경을 갖추고 있어야 하고새로운 기술을 연구하고 기술 트렌드 따라가는 데에 시간을 투자해야 한다.

빈번하고 단순한 업무는 위임하라

빈번하지 않고 단순한 업무는 직접 처리하라.

빈번하지 않고 복잡한 업무는 앞으로 리더가 될 사람을 교육하는데 사용하라.

빈번하고 복잡한 업무는 팀의 발전을 위해 위임하라.

개발 팀 운영의 건강도 확인하는 방법

- 릴리스 빈도
- 코드 체크인 빈도
- 문제 발생 빈도 (~ 미션 크리티컬 시스템)

개발자의 경력 경로는 사다리가 아니라 정글짐이다.