맨먼스미신 Chapter 2

일정 시간 부족은 모든 제품/소프트웨어의 실패의 주요 문제

1. 추정을 잘 못함

2. 진행 상황을 노력과 비교하고 이에 혼란이 옴.

3. 추정이 나쁘면 소프트웨어 관리자의 고집이 부족해진다.

4. 진행상황 모니터링 불량

5. 일정 유출 시 인력 추가(인력과 시간은 상호 교환 가능)

Optimism

낙관주의는 프로그래머들에게 가장 큰 자산이자 문제이다.

낙관주의는 아이디어화 단계에서 발견될 수 있는 반면 낙관주의는 구현과 상호작용에 따라 감소.
=>아이디어가 구현되어야 하기 때문.

아이디어의 "불완전성"과 "불완전성"
실행 매체를 다루기 쉽게 만드는 것은 소프트웨어를 적응시키는 가장 중요한 원칙.

Man-Month

노동력과 시간은 서로 바꿀 수 있는 것이며, 노력에 대한 잘못된 추정은 다른 이유로 인한 실패에 비해 또 다른 주요 문제이다.

1. 업무가 분할 가능한 경우, 인력 추가는 긍정적인 영향을 미침.

2. 분할할 수 없는 업무의 경우, 인력 추가는 "ZERO" 긍정적인 효과를 가져오지만 부정적인 효과만 가져옴.

3. 업무는 분할이 가능하지만 소통 요구사항과 같이 종속성이 있는 경우 효과는 여전히 긍정적이지만 시간의 추정은 증가.

4. 종속성이 얽혀 있고 상호관계가 더 많으면 [n*(n-1)]/2 만큼 필요한 노력이 증가. 이러한 상황에서, 더 많은 인력을 추가하는 것은 성공을 제외한 모든 부정적인 결과를 도출하고 결과물을 전달하는 데 필요한 시간을 늘림.

System Test

소프트웨어를 성공적으로 사용하기 위해서는 충분한 시스템 테스트 시간이 필요.

스케줄은
1. 1/3 - 계획

2.1/6 - 코딩

3.1/4 - 구성 요소 테스트 및 초기 시스템 테스트

4.1/4 - 전체 시스템 테스트를 통해 모든 구성 요소 확인

여기서 1/2 은 코드화된 소프트웨어를 테스트 및 디버깅하고 시스템을 테스트하는 데 사용되며, 주요 코딩은 필요한 시간의 1/6 을 제공.

계획을 세우는 데는 1/3 의 시간이 필요.

Gutless Estimation

고객 또는 후원자의 긴급성은 일정 부분에 큰 영향을 미치며, 이는 실제로 예정된 완료 계획으로 이어질 수 있지만 실제 완료 계획은 아님.

성공적인 결과를 전달하기 위해서는 적절한 계획이 필요하며 추정치가 전달되어야 함.

사고에 대한 보고서, 문제 해결에 필요한 시간 및 개발을 게시하면 이 문제를 해결하는 데 도움이 될 수 있음.

Regenerative Schedule Disaster

일반적인 개념은 프로젝트가 예정보다 늦을 때 더 많은 인력을 추가하는 것인데, 이는 잘못된 것임.

1. 마일스톤과 함께 제 시간에 작업을 완료한다는 가정

2. 이정표가 어긋나더라도 제시간에 임무를 완수한다는 가정

3. 일정 변경 - 전표 없이

4. 작업 다듬기

더 많은 인력이 추가할 때, 필요한 훈련 기간을 고려하지 않는 것은 좋지 않음.

잘못된 추정은 첫 번째 단계나 중간 단계뿐만 아니라 모든 단계에서 발생할 수 있습니다.

브룩의 법칙
"늦은 소프트웨어 프로젝트에 인력을 추가하는 것은 나중에 만든다."

필요한 인력은 독립적인 업무의 수에 따라 달라짐.

작업 수, 특히 독립적인 작업 및 인력 요구사항에 따라 일정을 설정할 수 있음.