게임소프트웨어공학 2차시험 정리

Lecture 3

**소프트웨어 개발의 위기 (원인)**

1. 소프트웨어의 대규모화
2. 유지보수의 어려움
3. 개발 일정 및 소요 예산 예측의 어려움

* 높은 수준의 소프트웨어가 요구될 경우
* 개발 일정 측정 어려움, 소요 예산 예측 불가능

1. 신기술 적용에 대한 방법론 부재

소프트웨어 개발의 위기 (결과) – 안 나올 가능성 높음

1. 프로젝트 예산 초과
2. 프로젝트 일정 지연
3. 낮은 품질
4. 유지보수 어려움

소프트웨어 개발 프로세스

1. 폭포수 모델 (시험 안 나옴)
2. 프로토타이핑 모델

* 전체 소프트웨어의 일부를 빠르게 제작하는 방식.
* 장점

: 요구사항을 정확하게 판단할 수 있음

실현 가능성을 빠르게 확인 가능

* 단점

: 완제품으로 오해

프로토타입 개발 내용이 완제품에 반영되어 품질 저하

1. 나선형 모델

* 개발 단계를 반복적으로 수행하여 점차적으로 안정적인

소프트웨어를 개발하는 방식

* 장점

: 큰 시스템을 안정적으로 개발할 때 효과적

순차적으로 검증이 이루어지기 때문에 안정적인 문제 해결 가능

유지보수와 개발의 일원화 가능

* 단점

: 많은 투자가 필요하여 작은 규모의 프로젝트엔 어울리지 않음

관리가 복잡함

Lecture 4

애자일 선언문 – 빨간 부분 빈칸 채우기로 시험 나옴

우리는 소프트웨어를 개발하고 또 다른 사람의 개발을 도와주면서 소프트웨어 개발의 더 나은 방법들을 찾아가고 있다. 이를 통해 우리는 다음을 가치 있게 여기게 되었다.

공정과 도구보다 개인과 상호작용을,

포괄적인 문서보다 작동하는 소프트웨어를,

계약 협상보다 고객과의 협력을,

계획을 따르기보다 변화에 대응하기를

가치 있게 여긴다. 이 말은, 왼쪽에 있는 것들도 가치가 있지만, 우리는 오른쪽에 있는 것들에 더 높은 가치를 둔다는 것이다.

Lecture 5

**왜 프로젝트가 실패하는가? (빈칸)**

- 소통의 부족

- 소통을 중요시하는 개발 방식 필요

**XP개요**

**-** 가치와 그 가치를 이루기 위한 실천법, 이 두 개를 큰 축으로 놓고 원칙을 세워서 그 둘 사이에서 균형을 맞추는 것을 권장함

**XP가치**

**1. 의사소통**

**2. 용기**

**3. 피드백**

**4. 단순함**

**5. 존중**

**스크럼**

- 특정 언어, 방법론에 의존적이지 않음

- 넓은 응용 범위의 기법

- 상호적, 점진적 개발

- 반복, 반복, 반복

- 문제를 해결하기 위한 프레임워크 개념

**스크럼 역할자**

**1. 제품 책임자**

- 제품 기능목록을 작성

- 개발 우선 순위 조정

- 개발 항목 추가

- 개발이 시작되면 팀 관여 최소화

**2. 스크럼 마스터(가장 중요한 인물)**

- 스크럼의 원칙과 가치를 기반으로 팀이 개발을 진행할 수 있도록 지원

- 팀의 업무를 방해하는 요소를 제거하는 것이 가장 큰 임무

**3. 스크럼 팀**

- 5~9명으로 구성

- 일정 기간 동안 기능 구현

- 구현해야 하는 기능을 완료하기 위해 노력하며 그에 따른 권한을 가짐

Lecture 6

**칸반 개요**

- 시각적 신호를 사용, 진행중 업무의 양을 제한하는 흐름 시스템

- 당김 시스템

**칸반 원칙**

- 워크플로우 시각화

- 진행중인 작업 제한

- 흐름 관리