**<시스템분석 중간고사 이론 대체과제>**

**12171661 윤혁**

**Prob. #1 [20 points] Explain the word, briefly.**

1. **SDLC**

- SDLC란 Systems Development Life Cycle의 약자로 시스템을 만들어 나갈 때 하는 일련의 과정입니다. 이 단계는 Planning, Analysis, design, implementation 으로 이루어져 있습니다. 이 단계들을 진행하면서 퀄리티가 높은 소프트웨어를 만드는 것이 목표입니다. 마지막 단계인 implementation에서는 시스템의 퀄리티를 높이기 위해 소스코드를 제공하고 소프트웨어를 완성 후 마무리 하는 데, 미비한 부분이 있거나 기존의 계획과 다른 부분이 있거나 기능이 누락됐다면 단계를 재회전합니다. 이 순서를 잘 지키지 않으면 시간과 예산이 절약된다고 생각 할 수도 있지만 퀄리티에 문제가 생길 수 있어 순서를 지키는 것이 중요합니다.

1. **UML**

- UML이란 Unified Modeling Language의 약자로 디자이너끼리 소통을 할 수 있는 언어인데, 일종의 설계도면입니다. 각각 회사에서 쓰이는 다른 언어들을 어떤 커다란 학회에서 여기저기서 같은 뜻으로 쓰인 모양들으 통합해서 하나의 언어로 만든 것입니다. 보통 Object-Orieted 컨셉에 따라 만들어지고 공통된 Vocabulary를 사용합니다. 프로그램을 디자인 할때 Analysis 부터 Implementaion 까지 그림으로 표현할 수 있습니다. 현재 UML의 버전은 2.0이고 여러 다이어그램들이 있는데 두 그룹으로 나누면 Structure diagram(소프트웨어의 구조에 중점), Behavior diagram(소프트웨어의 행동에 중점)이 있다.

**Prob. #2 [15 points]**

**Explain the difference between Parallel Development Methodology and Throwaway Prototyping Methodology.**

- Parallel Development Methodology와 Throwaway Prototyping Methodology의 차이점음 Parallel 방식은 design 단계부터 여러개의 프로젝트를 병렬로 과정을 거쳐서 implementation을 통해 하나의 결과물을 얻는 것인데 이것은 다시 뒤로 돌아가는 방식이 아닙니다. 곧, 피드백이나 요구사항을 수용하지 않고 여러개의 안을 하나로 통합하여 시스템을 만드는 것입니다. 반면에 Throwaway 방식은 Analysis 단계 이후에 Design과 Implementation으로 Design 프로토타입을 만들고 이 프로토타입으로 다시 과정을 거쳐 시스템으로써 기능이 가능한지, 마치 실험실에서 실험툴을 갖고 하는 것 과 비슷한 느낌으로 진행이 가능하다고 볼 수 있습니다. 그래서 리스크가 최소화 되고 실제 시스템이 만들어지기 전에 피드백과 지적사항들을 반영해 Design 프로토타입을 업데이트하여 최종 시스템을 만들 수 있습니다.

**Prob. #3 [15 points]**

**Explain the difference among Technical feasibility, Economic feasibility, and Organizational feasibility.**

- 첫번째로, Technical feasibility는 그 회사가 시스템을 제작할 수 있는가가 중요한 점이다. 소프트웨어 애널리스트가 예를들어 반도체 공정 소프트웨어를 제작한다면 반도체 공정도 잘 알고 있어야 할 것입니다. 그래서 그 분야와도 친숙해야하고 친숙할 수록 리스크도 적어집니다. 또한 사람, 기술 등 여러 방면에서 프로젝트의 사이즈, 비용 등이 커지게 되면 더 큰 리스크를 가져올 수 밖에 없습니다. 그리고 그 소프트웨어가 주변 환경. 즉 주변의 하드웨어 또는 소프트웨어 들과 잘 호환이 되어 문제없이 기능할 수 있는지도 기술적 요소중 하나입니다. 두번째로 Economic Feasibility는 그 소프트웨어를 제작하는데 드는 비용과 얻을 수 있는 이익을 계산합니다. 또한 미래가치도 예측하여 기대수익을 창출할 수 있어야 합니다. 그러한 비용과 이익의 흐름을 잘 캐치하고 있어야 합니다. 세번째로 Organizational Feasibility는 유저들이 이 시스템을 받아들일 수 있느냐의 문제입니다. 예를들어 SNS를 만든다 하면 이 소프트웨어의 보안성과 사람들의 사생활을 얼마나 침해하는지도 문제가 될 수 있을 것입니다. 아무리 기술적으로 훌륭하고 경제적으로 효율적인 시스템일지라도 대다수의 사람들이 인정하는 선을 지키고 그 범위 안에서 개발을 해야 할 것입니다. 시스템이 사회에 끼치는 영향, 현상 등을 Organizational management를 통해 파악해야 하며 유저들 그리고 유저가 아니더라도 다른 사람들에게 끼치는 영향과 여론, 사용자 의견을 꼭 중요하게 받아들여서 시스템 개발, 유지보수에 반영해야 좋은 시스템이 될 수 있을 것입니다.