Q/A Sheet #2 – Sociotechnical system

Date: 3/27 number: 2016311821 name: 한승하

Questions from Prof

1. Explain the strengths and weaknesses of the Waterfall model

   ⇨ Parallel development와 Easy of management는 Waterfall model의 가장 큰 강점이다. 하지만, Process이 어느정도 진행된 이후에 변경이 필요 해 졌을 때 이전 단계에 대한 변경이 어렵다. 또한 Customer가 요구하는 definition이 명확하지 않을 때 사용하기가 어렵다는 단점이 있다.

2. Explain the strengths and weaknesses of the incremental model

   ⇨ Definition이 명확하지 않은 Project에 대해 접근하기가 더 쉽다. 따라서 Customer requirement가 변경되었을 때, 수용하기가 훨씬 쉽다. 또한 고객 입장에서 Customer의 Feedback을 수용하는 것도 훨씬 더 유용하다. 또한 고객에게 빠르게 배포할 수 있다는 장점이 있다. 하지만 Incremental model은 초기 단계가 계속하여 반복되기 때문에 문서화하는 것이 굉장히 어렵고, 이는 Process가 visible하지 못하게 만들며, 이는 추후단계에서 문제가 될 수 있다. 또한 시스템의 구조가 새로운 Increment가 추가될 때 마다 망가지게 될 수 있다.

3. Look for refactoring concepts and find examples of them

   ⇨ Refactoring은 외부동작을 바꾸지 않으면서 내부구조를 개선하는 방법으로 개발을 하면서 지속적으로 좋은 디자인을 찾는 Incremental Model에 해당하는 concept라고 할 수 있을 것 같다. 목적은 Software을 보다 이해하기 쉽고, 수정하기 쉽게 만드는 것이며, 겉으로 보이는 기능은 변경하지 않는 것을 목표로 한다.

   Refactoring의 예시로는 Extract Method, Move Method, Introduce Explaining Variable등의 방법론이 있다.

4. Compare the waterfall model with the incremental model for the workflow

   ⇨ Waterfall model은 Requirement definition -> design -> implementation -> testing -> operation의 순차적 단계를 가지고 있으며, 단계를 돌아가는 것이 굉장히 제한적이다. Development의 과정에서 Requirement + design이 40%, implementation 20%, testing 40%으로 40-20-40 Rule을 이야기하기도 한다.

   ⇨ Incremental model은 이전 강의에서 Iterative model로 설명하였고, Outline -> Concurrent activities (Specification, Development, Validation) -> Final version의 순서를 가지며, 단계를 자유롭게 넘나들 수 있다.

5. Describe four fundamental process activities

   ⇨ Specification, development, validation, evolution 4가지를 말한다.

   Specification은 명세화로 고객이 가지고 있는 요구사항을 명확히 또는 시스템을 개발하는데 충족해야 하는 제약사항들을 명확히 하는 작업으로 이 작업이 잘못되면 이후 작업들의 의미가 없어진다.

   Development는 Designed and programmed로 Specification을 포함하기도, 하지 않기도 한다.

   Validation은 개발된 소프트웨어가 실질적으로 고객의 요구사항, 제약사항을 모두 충족하는지 확인하는 작업이며.

   Evolution은 많은 변화요인에 맞추어 시스템을 변화시켜가는 과정을 이야기한다.

6. Describe the requirement engineering process with key activities and their work products

   ⇨ Feasibility Study는 타당성 조사로, 비즈니스 측면에서 필요한가, 기술적으로 가능한가, 법적으론 문제가 없는가, 등에 대한 조사를 진행하며, 가치가 있고 문제가 없음을 검토하는 과정이다. 이의 결과물로써 Feasibility report 즉 타당성 조사 결과서가 나오게 된다.

   Requirements elicitation and analysis과정은 다양한 Stakeholder들에게 요구사항을 추출해 오는 작업을 말하며, 분석을 통해 반영시킬 Requirement들을 추출해 낸다. 이의 결과로 System model이 나오게 된다.

   Requirement specification은 명세화로써 User and system requirement를 결과물로

나오게 된다. 이는 고객과 개발자들이 이해할 수 있도록 개발 과정을 명세화 하는 것을 말한다. 이의 결과물로 요구사항 명세서가 나오며, 이는 이후 법적인 공방에서도 꼭 필요하다.

Requirement validation 과정은 요구사항을 확인하는 작업으로, 이 과정에서 문제가 발견되어 이전단계로 넘어가는 Cost는 높지 않지만, 이후 Implementation으로 넘어갈 경우 다시 돌아오는 Cost가 굉장히 높기 때문에 요구사항을 자세히 확인하는 것이 중요하다. 이의 결과로 Requirement document, Requirement specification이 나오게 된다. 이를 계약 문서라고도 한다.

7. Describe the differences between component, system and acceptance tests.

   ⇨ component는 각 Component개발자에 의해 개별적으로 Testing이며, unit testing, module testing이라고도 한다.

   System testing은 Whole system단위로 system data에 인한 testing되며, 이로 인해 emergent properties에 대한 첫 점검이 이루어진다.

   Acceptance testing은 customer data로 고객의 환경에서 testing이 이루어지며, 인수시험이라고 하기도 한다.

8. Slide on page 28, discuss when to plan each test with why

   ⇨ Acceptance test는 고객의 환경에서 testing을 해야 하기 때문에, 초기에 고객의 Requirement를 specification 할 때, 이에 대한 testing plan을 같이 세워 놔야 이후 완성된 system이 초기 목적을 잘 만족했는지 validation이 가능하다.

   이와 유사하게 system integration test, sub-system integration test또한 각각 system specification, system design과 함께 계획이 수립되어야 하는데 이는 위의 Acceptance test와 동일하게 개발 이전에 목표를 명확히 해야 하는 것과 일맥상통한다고 할 수 있다.

9. Change is inevitable in every large software project. There are two related approaches you can use to reduce rework costs. First, explain the concept or rework and describe two approaches

   ⇨ Rework은 변경요청에 대해 Requirement analyzing, redesign, reimplement, retesting의 과정을 다시 밟는 것을 말한다. 이의 cost를 줄이는 방법은 2가지로

Change avoidance, Change tolerance가 있다.

Change avoidance는 prototype system을 개발하여 고객에게 조금이라도 빨리 system을 보여줘서 feedback을 받아 불명확한 요소를 최대한 없애려고 하는 방법이다.

Change tolerance는 incremental development이라고 이야기할 수 있다. 개발과 testing을 지속적으로 incremental방식이 가능하다면 change에 대한 feedback이 훨씬 수월하기 때문이다.

10. Discuss the benefits of prototyping. And explain the reasons for improved maintainability on page 38 of the slide.

    ⇨ Prototype은 user의 needs을 맞춰주는데 용이하고, design quality, maintainability를 높이며, development effort를 줄여준다. 이는 prototype이 고객의 requirement를 좀 더 명확히 할 수 있다는 점에서 모든 development 과정에서 error혹은 change의 가능성을 명확히 줄여준다 따라서 maintainability의 경우에도 prototype을 통해 requirement가 더 확실한 개발이 된다면, 이전에 설명했던 rework의 cost를 줄일 수 있기 때문이다.

11. Compare throw-away prototyping with exploratory prototyping

    ⇨ Throw-away방식은 prototype는 어디까지나 requirement를 명확히 하기 위한 과정으로, 이것이 만족되었을 경우 prototype을 버리는 것을 말하며

    Exploratory방식은 초기 Prototype에 조금씩 기능을 확장해 나가며 system을 구성하는 방식이다

12. With incremental delivery, customers can feel more confident about the system. Guess why

    ⇨ Incremental delivery는 매 개발 작업마다 customer에게 결과치를 보여줌으로써, 고객이 개발이 어느정도 진행되었는지에 대한 정보를 얻을 수 있으며, 매 delivery마다 요구한 requirement에 대한 확인을 직접 확인할 수 있으므로 development에 대한 confident를 가질 수 있다.

13. Complete testing is virtually impossible. Discuss cost-effective testing strategies in development environments where time and cost are inadequate. Hint: Take advantage of available statistics.

    ⇨ Testing의 cost를 줄이는 방법은 기본적으로 유지보수의 rework cost를 줄이는 방법과 굉장히 유사할 수밖에 없을 것으로 추정된다. 이는 testing의 목적이 change을 줄이는 것이기 때문이다.

    따라서 이전에 change avoidance, change tolerance방법으로 제시되었던 모든 방법들이 testing의 cost를 줄이는데 사용할 수 있다.

    대표적으로는 Prototyping, Increment development, Increment delivery들을 이야기할 수 있을 것 같다.

Questions of yourself

1. Coder와 Software Engineer의 차이는 무엇인가?

    ⇨ Programmer, Coder는 앞서 주어진 Specification, Design이 주어졌을 때 이를 코드화 시킬 수 있는 사람을 말하며, Software Engineer는 Specification, Design 즉 Requirement Engineering부터 코드화까지 가능한 사람을 말한다.

2. Back-to-back testing에 대해 기술하라

    ⇨ 다양한 case에 대해 상세한 flow를 확인하는 것이 아닌 입력 요소에 대한 output을 확인하여 판단하는 것을 말한다.