QA Sheet 4

2017310695 이주민

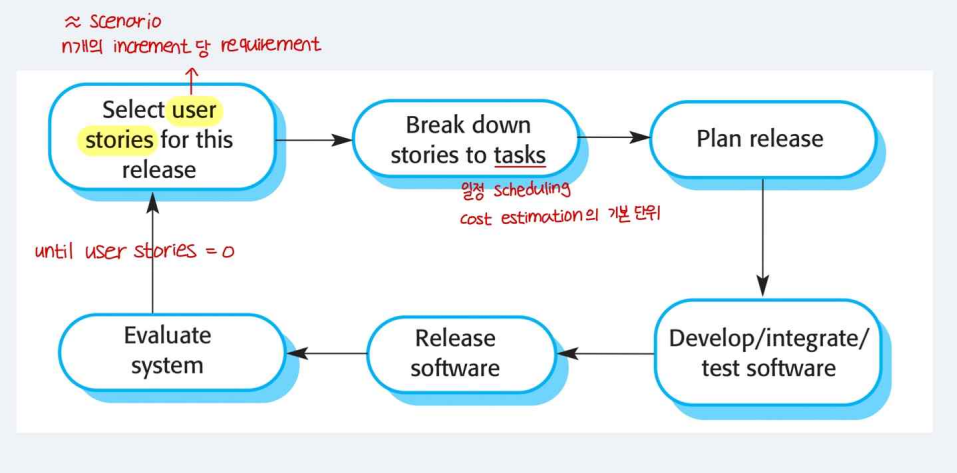
1. Discuss and write your own ideas on why rapid development and delivery is important from a business and technology perspective.
: 현대의 business와 techique는 경쟁사의 새로운 제품 출시/소프트웨어 요구사항 변경 등 굉장히 빨리 변화하는 시장이므로 빠른 개발, 배포를 통해 고객의 need를 충족하는지 test를 받는 것이 중요하다고 생각한다.

2. Explain the principle of agile methods with each reason.
① Customer involvement : 개발 과정에 있어서 customer가 개발 팀의 구성원으로서 적극적으로 참여해야 한다는 것으로 새로운 system requirement를 제공하고, 우선순위를 정해주며 진행중인 system을 평가하는 역할을 맡음
② Incremental delivery : 점진적 개발과 배포를 통해서 cutomer의 need를 잘 충족하고 있는지 계속 test, review 받는 것
③ People not process : process보다는 개발자, 개발 팀이 중요시 된다는 것으로 이들의 역량이 최대로 발휘되기 위해서 그들 간의 interaction이 중요시 됨
④ Embrace change : system requirement는 변화할 수 있기 때문에 이 변화를 수용하고 system에 반영
⑤ Maintain simplicity : 코드의 복잡성을 없애고 간단히 유지하여 document가 적은 agile method의 maintenance를 높임 by refactoring

3. Discuss the strengths and problems of agile methods.
- 장점 : incremental하게 소프트웨어를 개발하므로 문제점이 발생하거나 요구사항이 변경되어 수용해야 할 경우 다음 increment에서 반영이 가능하여 plan-driven method보다 유동적이게 대응할 수 있음. 또한, 코딩과 customer과의 interaction에 집중하여서 필요한 문서의 양을 최소화 시킬 수 있어서 소규모 시스템에 적합함.
- 단점 : 실제로 customer의 지속적인 clear commitment를 받기가 어려움. 또한 여러 명의 stakeholder들의 모든 이해관계를 파악하기 어려우며 이에 대한 우선순위를 매기는 것도 어려워서 stakholder들의 review에 대해 적절하게 응답하는데 어려움.

4. Summarize the determinants when deciding on a plan-driven or agile approach to system development.
1) plan-driven
- large system으로 자세한 명세서와 설계를 필요로 하는 경우
- 개발 전에 많은 분석이 요구되는 시스템
- long-liftime system의 경우 많은 설계 문서들이 필요하므로
- 개발 팀이 distributed되어 있거나 개발의 일부를 outsourcing 할 경우, 의사소통을 위해
- 문화적이나 조직적 이슈에 영향을 많이 받는 경우
- 개발자의 skill이 높지 않다면, 자세히 설계되어 있는 것을 code로만 바꾸면 됨
- 외부 규제 사항이 많은 시스템의 경우 그 규제에 관한 자세한 문서가 필요함(system 전체 관점에서 외부 규제 사항을 대응하기 때문에 agile 방법론에서는 이를 대응하기 어려움)
2) agile
- software를 배포했을 때 customer로부터 빠른 피드백을 얻어야 하는 경우
- small or medium system인 경우

5. Discuss the release cycle of XP and compare it to the waterfall model.



: XP의 첫 번째 단계는 많은 user story 들 중에 한 가지를 선택하는 것이고 두 번째로 선택한 user story를 task 단위로 쪼개는 일을 말함. 세 번째는 이렇게 나눠놓은 task별로 plan을 수립하고 네 번째로 task별로 개발, integrate, test하는 단계를 거침. 마지막으로 소프트웨어를 release하게 되고 system을 평가 받음.

XP의 한 release cycle은 waterfall 과정과 비슷하지만 모든 과정이 끝나고 한번에 system을 relase하는 waterfall과는 다르게 XP는 한 cycle이 끝날 때마다(user story 하나 당) release를 하여 customer의 평가, review를 받고 다음 cycle에 이를 반영할 수 있음

6. Summarize the important features on practices of XP.

- TDD : test code를 작성하고 그에 해당하는 production code를 나중에 작성하는 방식으로 test code를 작성하기 때문에 application을 테스트할 때 자동화 할 수 있다는 장점이 있음

unit test로 unit의 시스템이 복잡할수록 이익이 높아짐 내가 개발하는 코드의 특징을 이미 알고 있어야 함(어떤 input을 넣었을 때 어떤 output이 나올지 이미 알고 있어야함)

- Pair Programming : driver와 observer로 짝을 지어 개발하여 서로의 작업을 체크해주는 장점이 있으며, Collective owenership을 가지게 해준다는 장점이 있음
- continuous integration : daliy cycle로 어제까지 한 작업과 오늘 한 작업을 합치는 작업

7. Read the story card on slide 22 carefully and look for tasks 4 or above.

: prescribe한 것을 audit database에 기록할 것인지 다시 변경을 할 것인지.

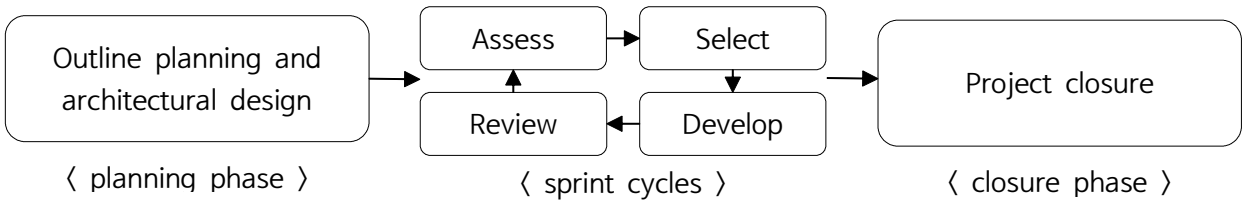
8. Explain test-driven development and think about the impact on SW quality.

: 구현할 requriement들을 더 명확하게 알 수 있으며 테스트가 데이터가 아니라 프로그램을 통해 진행되기 때문에 새로운 기능이 추가되었을 때 자동으로 이전에 있었던 것과 새롭게 추가된 테스트들이 자동적으로 진행되어 regression test가 쉬워짐

- software quality란 software의 버그가 적고 빠른 시간 내에 처리될 수 있는 것 reliabliity safety 등등
- 버그를 일찍 발견할 수 있음.
- 잘 짜여진 구조와 함께 확장성이나 코드의 간결성을 얻음 (test-driven development이므로 test case를 실행시킬 수 있는 test code를 미리 작성하여 input, output를 가지고 있으니 그거에 맞는 코드를 짜므로 불필요한 코드를 만들지 않음)
- test를 무조건 하고 넘어가기 때문에 좋고, 우선순위가 높은 모듈은 테스트가 시간에 지남에도 계속해서 반복하게 되므로 고객들이 좋아함

9. Summarize the advantages of pair programming.
: 매일 observer와 driver 짝이 바뀜으로서 개발팀의 한 사람이 떠나도 전체 팀원이 코드에 대해 알고 있으므로 project에 문제가 발생하지 않음(개발팀 모두가 system code에 owenership과 책임감을 가지게 됨). 또한 observer가 driver의 코드 작성을 지켜보므로 informal review의 역할을 하게 되며, refactoring을 동시에 진행할 수 있음.
- code review : 시간이 걸림, 다른 사람의 코드를 리뷰하다는 것 자체가 challenging함
- 하지만 pair programming은 즉각적으로 가능하므로 code review의 상위 버전임 + 코드가 다 작성되기 전까지 볼 수 있어서 버그 발견에 쉬움.

10. Explain the SCRUM process on slide 40.



① Outline planning and architectural design phase
: 여러 story cards를 작성하고 system의 architectural design을 하고 목표를 설정함

② Sprint cycles
: 한 story 당 기간은 2~4주정도로 너무 짧으면 부하가 커지게 되고 너무 길면 관리가 힘들어지기 때문에 2~4주를 지키는 것이 좋음. Access 단계에서는 product backlog(프로젝트에서 수행해야 할 작업들의 리스트)를 작성하는 것으로 시작하여 이를 Sprint backlog로 나눠 이 sprint cycle동안에 수행해야 할 작업을 계획한다. Select 단계에서는 고객과 이 sprint cycle동안 어떤 특징과 기능을 개발할 것인지 선택함. Develop 단계에서는 개발팀은 더 이상의 수정, 변경을 줄이기 위해 고객으로부터 고립되는데 이때, Scrum master를 두어서 그를 통해 외부와 의사소통을 함. Review단계에서는 고객, 이해당사자와 함께 평가, review함. 이 네 가지 과정을 모든 story cards를 다 할 때까지 반복함.

③ Closure phase
: system help frame이나 user manual같은 문서들을 작성하고 프로젝트를 마무리함

11. Summarize the benefits of the SCRUM.
: agile의 단점은 문서와 소통이 부족하다는 단점이 있는데, scrum은 매일 소통하고 backlog를 통해서 문서가 생성이 되므로 agile을 보완할 수 있음
- Product를 이해 가능한 조각(sprint backlog)으로 나누어 관리할 수 있음
- 요구사항이 불안정해도 project의 진행을 할 수 있음
- 개발팀 전부가 project에 대해 정확히 알고 있어서 팀 간의 communication이 원활하게 이뤄질 수 있음
- 고객 입장에서 increment가 개발될 때마다 볼 수 있기 때문에 제품에 대한 feedback을 줄 수 있음
- 이러한 점에서 고객과 개발자 사이에 믿음이 생기고 모든 사람이 project의 성공을 기원하는 긍정적인 분위기가 형성될 수 있음

12. Summarize the strategies for scaling up and scaling out of agile methods.

① Scaling up

: agile method를 상대적으로 규모가 큰 소프트웨어 개발에 적용하는 것

- 이를 위해서는 코딩에만 집중할 것이 아니라 구체적인 디자인과 문서화가 선행되어야 함.
- 규모가 큰 시스템 개발이기 때문에 팀 간의 의사소통 방법이 잘 고안되어야 함(ex) 화상회의로 개발과정에서 업데이트된 사항들을 숙지)
- 큰 규모의 시스템에 대해 주기적으로 integrated된 소프트웨어를 개발하여 release해야 함

② Scaling out

: agile method가 어떻게 큰 조직에 적용될 수 있는가 하는 것

- Agile method에 대해 익숙하지 않은 프로젝트 매니저들은 이 새롭지 않은 방법에 대해 꺼려 할 수 있음
- 큰 조직의 관료적인 특징과 여러 규칙들 때문에 혹은 agile method와는 맞지 않는 그 조직만의 전통적인 개발 방법을 이용해 왔을 경우 agile method와 맞지 않음
- agile method는 상대적으로 높은 기술을 가지고 있는 팀 멤버들에 적용할 때 큰 효율을 내지만 큰 조직에서는 능력이 모두 일치하지 않기 때문에 적용하기 어려울 수 있음

MY QUESTION

1. TDD에서 test case가 아닌 test code를 작성하기 때문에 application을 테스트할 때 자동화 할 수 있다는 장점이 있다고 하는데 test code를 작성하면 왜 이런 장점이 생기는지?

test case : data(input, expected output)

application code를 평가하기 위해서 test case를 사용하는데 규모가 큰 소프트웨어를 개발하면 개수가 많아짐. test code는 test case와 또 하나의 input인 application code를 받아서 이를 실행시켜 이의 output을 expected output과 비교하는 것.

따라서 어떤 테스트 데이터가 추가가 되도 test를 할 수 있음

2. Agile method에서 여러 명의 stakeholder들의 모든 이해관계를 파악하기 어려우며 이에 대한 우선순위를 매기는 것도 어렵다고 했는데 우선순위는 어차피 customer들이 매기는 것이 아닌지?

: customer 내의 stakeholder 끼리 원하는 것이 다르기 때문에 그들끼리 우선 순위를 정하는 것이 어려움,