Q/A Sheet #5 – Sociotechnical system

Date: 4/17 number: 2016311821 name: 한승하

Questions from Prof

1. Define the notion of requirement and requirements engineering process

   A. Requirement는 개발하는 System이 제공해야하는 Service에 대한 Description이며, 또한 내, 외부에 존재하는, System을 운영해 나가는데 충족해야 하는 제약사항들에 대한 Description이다.

   Requirement는 한가지 Level이 아닌 high-level abstract statement에서의 디테일한 mathematical functional specification을 필요로 한다. 이는 개발에 참여하는 사람들이 내용을 쉽게 보고 이해할 수 있게 하는 목적과, 계약에 있어서 요구사항을 명확히 하게 하기 위한 목적을 위해서 이루어진다.

   Requirement Engineering은 이러한 Requirement들을 확립하는 과정을 의미한다.

2. Compare user requirement and system requirement

   A. User requirement는 Natural language 와 Diagrams를 사용하여 System이 제공하여야 하는 services 들을 기술하며, 이는 Customer를 위해 중요한 정보로 작용한다. Client Managers, Contractor Managers들이 User requirement의 가장 주요 Target이 된다.

   B. System requirement는 User requirement를 상세히 기술한 것으로써, 일정한 체계화 되어지고 구조화되어진 form을 통해 작성하는 것이 특징이다. System requirement는 개발자들에게 중요한 정보이며, client와 계약을 위한 중요한 정보가 된다. Software Developers는 System requirement의 가장 중요한 Target이다.

3. Compare functional requirement. Nonfunctional requirement and domain requirement

   A. Functional requirement는 기능 요구사항으로, System이 제공해야하는 Service에 대한 기술, 특정 input혹은 환경, 상황에서의 output혹은 동작을 명세화 하는 것으로써, System이 작동해야 하는 방향, 혹은 System이 해서는 안되는 동작들을 말한다. Type of software, expected users, type of system에 영향을 받는다

   추상도 Level에 따라 User requirement와 System requirement로 분류할 수 있다.

B. Non-functional requirement는 비 기능 요구사항으로 각종 내 외부적인 제약사항들과 safety, performance와 같은 System properties와 같은 사항들을 말한다. 이는 IDE, Programming language, development method, 등도 포함될 수 있다.

   Non-functional requirement는 Functional requirement 보다 critical 하게 작용하며, requirement가 모두 충족되지 못한 경우 교섭의 가능성이 적어진다.

   Non-functional requirement는 전체적인 system의 구조에도 영향을 미치므로 초기에 충분히 검토되지 않는다면 치명적인 문제로 작용할 수 있다.

   또한 하나의 Non-functional requirement는 여러 개의 functional requirement를 불러올 수 있다.

4. Suppose that your client has a specific request for non-functional requirement such as high performance. And you have to determine the process model of the waterfall or agile model. Select a process model and explain why

   A. 이는 다른 Requirement들이 어느정도 명세화 되었는지에 따라 달라질 수 있을 것 같다. 일반적인 개발상황에서 Functional requirement 또한 평균 이상의 정도로 충족할 수 있다고 가정되었을 때, Water fall model을 선택할 것 같다.

      이는 Non-functional requirement가 많은 경우에서 명확히 하기 어려우며, 이를 specification하는데 많은 feedback이 필요로 하는데, 이러한 경우 Non-functional requirement가 명확히 기술되어져, Requirement specification이 보다 쉽게 진행될 수 있을 경우, Waterfall model이 더 많은 system 개발에 대해 범용적이고, 쉽게 적용 가능하며, parallel development 또한 더욱 유용하기 때문이다.

5. Explain why domain requirements are crucial for system development and difficult to articulate

   A. Domain requirement는 Environment의 영향을 받는 requirement로, 고객이 명확하게 제시해주지 않는 requirement이다. 따라서 개발자가 개발하는 System의 Domain을 학습하여, 도출해내는 과정이 필요하다. 이것이 어려운 이유는 각 영역에 대한 여러 용어, 기술, 등이 있고, Customer또한 개발자가 이것을 알 것이라 생각하여 자세히 기술을 꺼리기 때문이다.

      Domain requirement가 정확하지 않으면 개발된 System은 해당 환경에서 사용할 수

없기 때문에 Domain requirement가 매우 중요한 요소가 된다.

6. Discuss the requirement engineering process in the agile methods

   A. Agile method는 requirement specification 과정이 단발성으로 존재하지 않으며, 이에 따라 문서에 대한 중요성이 떨어지며, 문서화 보다는 수정, 변경에 용이한 코드를 작성하는 것이 중요하게 여겨진다. 따라서 agile method 지지자들은 requirement document를 작성하는 것을 시간낭비라고 여기는 경우가 많다.

      Agile method는 incremental requirement engineering 방식을 사용해 User stories단위로 requirement를 관리하며, 이를 여러 task들로 brake down되어 development되어진다.

      이는 현장에서 실제로 사용할 수 있는 방법이지만. 대규모의 system 개발, 규모가 큰 organization, requirement document가 중요한 임베디드 시스템, 펌웨어 등의 개발에서는 사용하기 어려울 수 있다.

7. Read and learn the form of requirement documents proposed by IEEE.

   A. Preface

      i. 예상되는 독자에 설정과 개발하는 System의 version의 역사와 새로운 version에 대한 요약 및 각 version에서 존재했던 변경점을 기술한다.

   B. Introduction

      i. System이 필요한 이유를 기술한다. System의 functions에 대한 간략한 기술과, System이 다른 system들과 어떤 식으로 사용되어질 지를 기술한다. 또한 system이 해당 software를 사용하는 단체의 business 혹은 전략적 목표에 어떻게 부합하는가를 기술한다.

   C. Glossary

      i. 해당 document에서 사용하는 기술적인 용어들에 대한 설명을 기술한다. 독자에 대한 경험 혹은 능력치를 가정하여 기술해선 안된다.

   D. User requirements definition

      i. User에게 제공되는 services들을 기술한다. Non-functional system requirement들 또한 기술되어야 한다. 자연어, diagrams, 혹은 기타 notations들을 사용하여 customer가 이해할 수 있게 기술되어야 하며, 충족되어야 하는 product, process

standards 또한 기술되어야 한다.

E. System architecture

   i. 전체적인 system architecture에 대한 high-level overview가 기술되어야 한다. 이는 전체 system modules에서 functions들의 분포를 담고 있어야 한다. 재사용되는 Architectural components들은 반드시 강조되어야 한다.

F. System requirements specification

   i. Functional, Non-functional requirements을 보다 자세하게 기술하여야 한다. 필요하다면 nonfunctional requirement들에 더 자세한 내용이 들어가도 된다. 타 system과의 interfaces 또한 기술되어야 한다.

G. System models

   i. Graphical system model를 사용하여 system components과 system, system environment들의 관계를 보여주어야 한다. Object model, data-flow model, semantic data models같은 예시가 있다.

H. System evolution

   i. System이 근반으로 잡고 있는 fundamental assumptions들에 대한 기술이 필요하다. 또한 hardware evolution, changing user needs, 등으로 인해 예상되는 변경점들 또한 기술되어야 한다. 이는 system designers들에게 용이하며, 후에 수정이 필요할 수 있는 결정들을 피할 수 있는 근거가 된다.

I. Appendices

   i. 개발되어지는 application에 연관된 자세한 정보를 기술해야 한다. 예시로 hardware와 database에 대한 설명, system에 필요한 hardware requirement에 대한 구성, system에 사용되어지는 data에 대한 Database requirement의 논리적 구성, data간의 relationship등이 있다.

J. Index

   i. Document의 index를 기술한다. 일반적인 alphabetic index와 diagrams에 대한 index, functions에 대한 index또한 포함된다.

8. Describe the requirement engineering process in the agile methods

   A. Agile method에서는 requirement engineering에 있어 customer의 참여가 필수적이며, Requirement가 Scenarios혹은 User stories의 형태로 관리된다. Incremental

planning단계를 거치며 기록된 story cards중 개발할 Story card를 고르고, 이를 Task 단위로 쪼개는 과정이 진행된다.

9. Explain the spiral view of the requirement engineering process

   A. Business requirement specification ~ Review의 과정을 Spiral 형태로 나타낸 것으로, Requirement elicitation, Requirements specification, Requirement validation, System requirement document의 4가지 작업을 Requirement engineering process을 거치며 반복적으로 수행하게 된다는 것을 말한다.

10. Summarize the process of requirements elicitation and the problems that occur during the process

    A. Requirement elicitation에선 Requirement를 획득하게 되며, requirement discovery, gathering, acquisition, 등의 다양한 표현을 사용한다.

    이 과정에서 application domain, system이 제공해야 하는 services, 내, 외부적인 constraints들을 명확히 하여 모으는 작업을 말한다.

    Managers, Maintain engineers, Domain experts, Trade unions등의 다양한 계층의 stakeholders들에게 요구사항을 획득하는 작업이 필요하다.

    Discovery, classification and organization, prioritization and negotiation, specification 로 세부화 할 수 있다.

    발생할 수 있는 problem들은 Stakeholders는 그들이 정말 필요하는 요소를 명확히 알지 못하며, requirement를 표현하는 것에 있어 그들의 background 혹은 지식을 이용한 언어로 표현하기에 이해가 어려울 수 있다. 또한 stakeholder의 spectrum이 넓을 경우 conflict가 발생할 수 있으며, Organization과 political한 영향을 받을 수 있다. 또한 분석 과정에서도 requirement가 변경될 수 있다는 위험요소를 가지고 있다.

11. Interviewing is a popular and effective way to gather user requirements. But there are also strengths and weaknesses. Explain them

    A. Interviewing의 장점은 stakeholder들이 system에 대한 어떤 기대가 있고, 어떤 요구사항이 있는지에 대한 overall understanding에 굉장히 효과적이다.

    하지만 Interviewing은 domain requirement에 대한 이해에는 좋지 않다.

Requirement engineer들이 Domain에 대한 지식을 전부 알 수는 없기 때문에, 해당 Domain에 대한 지식을 모두 알고 있는 stakeholder과의 interview를 통해 requirement를 모두 획득하는 것은 사실상 불가능하다.

또한 Interview가 장시간 진행되는 경우에 interviewer들이 집중하는 것이 어렵기 때문에, 주의를 환기시킬 수 있는 Springboard question들이 필요하다.

12. Describe a scenario about a specific service in your team project according to the template. And extract actors and use cases for each actor

    A. 팀 프로젝트 윤곽이 잡힌 이후 Q/A Sheet에 덧붙이도록 하겠습니다.

13. Describe the advantages and limitations of ethnography. And suppose its effective use situation

    A. Ethnography는 "참여적 관찰법"으로 이야기할 수 있다. Ethnography는 Stakeholder들이 Interview등의 방법으로 모두 알아내지 못한 Requirement를 획득하는 것에 효과적인 방법이다.

    Ethnography는 특히 System에 관련된 Social and organization factor들을 관찰하는데 굉장히 효과적인 방법이다.

    Ethnography는 system modeling으로는 표현하기 어려운 다양한 복잡한 내부의 내용들을 표현하는데 도움이 될 수 있다.

    Ethnography stakeholder의 existing process를 이해하는 것엔 좋은 방법이지만 개발되어져야 하는 new feature를 식별하는 것을 목적으로 할 수 없다.

    Ethnography를 사용하여 좋은 효과를 얻을 수 있는 사례로는 수업에서 언급한 Air traffic같은 system에 대한 관찰, 또한 이번에 터진 배달의 민족 사건에 대한 해결책 중 하나로 떠오르고 있는 지자체의 배달 system 개발과정에서 기존 system의 사용된 방법들을 분석하는 것에 효과적으로 사용될 수 있는 방법일 것 같다.

14. Define the process of requirement validation and explain its importance

    A. Requirement validation customer가 실제로 원하는 것과 requirement가 정의하는 것이 일치하는지 검증하는 과정이다. Requirement error cost는 이후의 error들 보다 훨씬 큰 cost를 가지고 있기 때문에 굉장히 중요하다.

또한 system의 결함 중 가장 큰 원인이 Requirement error이기 때문에 Requirement validation이 굉장히 중요하다.

Requirement validation은 Validity, Consistency, Completeness, Realism, Verifiability의 Check point를 가진다.

Validity: 개발되어지는 System의 function들이 customer의 need들을 최대로 support 가능한 형태인가? 에 대한 타당성, 유효성 검증이다.

Consistency: Requirement conflict가 없는가? 에 대한 일관성 검증이다.

Completeness: Customer가 요구하는 모든 function을 커버하고 있는가? 에 대한 완전성 검증이다. 이는 필요한 모든 function이 들어가 있는가? 에 대한 검증으로 function개별이 need를 최대치로 support하는가에 대한 Validity와 구별된다.

Realism: 모든 Requirement들이 available한 예산, 기술로 만들어질 수 있는가? 에 대한 실현가능성 검증이다.

Verifiability: 모든 Requirement들을 검정 할 수 있는가? 에 대한 확인/검정 가능성 검증이다.

이러한 Checking에는 Review를 주로 이용하며, 조직에 여유가 있을 경우 Prototyping을 또는 Test-case generation의 방법을 이용하여 검증 과정을 거친다.

15. List the requirements validation techniques and explain what you can identify with each technique

A. Requirement Review

i. Requirement 문서에 대한 Review를 진행한다. Formal review와 informal review 로 나눠질 수 있다. Requirement와 외부제약에 대한 검토를 통해 Consistency에 대한 검증이 가능하며, 고객주도의 Review 과정을 통해 Verifiability, Comprehensibility 즉 문서가 이해하기 용이한가, Traceability 즉 추적가능성에 대한 검증으로 Requirement를 요구한 stakeholder가 누구인지에 대한 검증, Adaptability 즉 Requirement에 대한 변화가 얼마나 큰 영향을 줄 수 있는가에 대한 검증할 수 있다.

B. Prototyping

i. 조직에 여유가 될 경우 사용할 수 있다. 초기모델을 사용하여 Requirement 충족 여부를 검토하는 것으로, 실제 초기 완성본을 배포하고 feedback을 받는 과정을 통해 모든 Checkpoint에 대한 검증이 가능할 것이다.

C. Test-case generation

i. Input, Output set의 testcase가 만들어 지는가에 대한 검증으로 Verifiability에 대한 검증이 가능하다.

16. Describe the process of requirement change management

A. Requirement management는 모든 과정에서 Requirement에 대한 변경이 요구되었을 때 이를 관리하는 모든 Process를 통칭하여 이야기한다. 각각의 Requirement를 구분, 식별, 추적하는 것이 필요하고, dependency에 대한 map을 명확히 가지고 있는 것, Change request를 처리하기 위한 formal한 process를 가지고 있는 것이 중요하다.

i. Requirement planning:

1. Requirement identification: Requirement 하나하나를 개별적으로 식별해야 한다.
2. Change management process: 변경에 대한 process가 정립되어 있어야 한다.
3. Traceability policies: Requirement 사이의 Traceability를 확보해 놓아야 한다.
4. Tool support: 변경 작업을 지원할 수 있는 Tool들이 명확하게 정립되어 있어야 한다.

ii. Problem and change specification analysis

1. 변경 요청이 충분히 필요한 Request인지를 검증한다.

iii. Change analysis and costing

1. 요구된 변경작업이 예산적, 기술적으로 실현 가능한지에 대해 검증한다 이 검증단계까지 거치면 Change implementation이 가능하다.

iv. Change implementation

1. 실제 Requirement를 변경하는 작업이다.

Questions of yourself

1. Non-functional requirement가 더욱 critical 한 이유는 무엇인가?

   A. Non-functional requirement는 System 전체적인 구조에 영향을 미칠 수 있다. 따라서 Non-functional requirement의 불명확한 시작은 이후에 System에 큰 문제를 야기할 수 있다. 또한 Functional requirement는 충족하지 못한 requirement이 생겼을 때, 협상의 여지가 있지만, Non-functional requirement의 부재는 사용할 수 없는 문제를 가져오기 때문에 협상 또한 어렵다.

2. Natural language의 한계는 무엇인가?

   A. Natural language는 해석적인 한계가 존재한다. 해석하는 사람에 따라 애매모호함이 있을 수 있으며, 다양한 해석이 가능하며, 아주 정확한 기술이 어렵다는 한계를 가지고 있다. 따라서 Structured natural language, design description language, Graphical notation, 등의 추가적인 작성기법의 사용이 필요하다.

   Natural language로 requirement가 작성될 경우, Customer는 같은 requirement에 대해 더 높은 기대치를, 개발자는 더 낮은 기대치를 가지는 경향성이 존재한다.