Q/A Sheet #2 – Sociotechnical system

Date: 3/21 number: 2016311821 name: 한승하

Questions from Prof

1. Compare sociotechnical systems to conventional technical computer-based systems

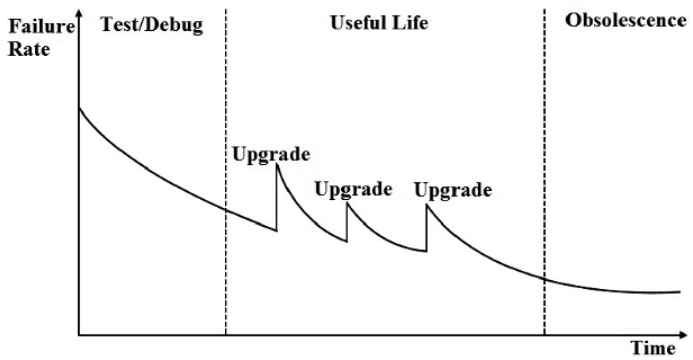
⇨ Technical computer-based systems는 Hardware와 software을 포함하지만 operators와 operational process를 system의 일부로 고려하지 않는다.

Socio-technical systems는 Technical computer-based systems의 요소들에 더해 Operational process나 technical system과 상호작용을 하는 사람들 까지도 system의 일부로 포함한다. Socio-technical systems는 주로 organizational policies 혹은 규칙을 따른다.

2. Assume the difference between a hardware failure and a software failure. Note that a hardware failure is represented by a bathtub curve

⇨ Hardware failure 곡선은 초기에 많은 failure을 일으키다, 상당한 시간이 지난 후에는 System이 안정화되어 failure가 점차 줄어든다. 하지만 Hardware lifetime이 지나는 시점부터 급격하게 증가한다 따라서 Bathtub curve라고 지칭한다.

Lifetime이 없는 Software의 Failure rate는 Hardware와 유사하게 Testing / 초기 상태에서의 failure가 높은 빈도로 나타내나 System이 안정화 됨에 따라 failure이 줄어간다. 하지만 hardware와 다르게 software는 지속적인 update/upgrade 과정을 거치므로 변화가 있는 시점마다 Testing / 초기상태를 거치게 되고 이는 높은 failure을 야기한다. 이후 System이 더이상 Update / Upgrade 가 진행되지 않는 상태가 되면 Failure는 점차 줄어들어 안정화 될 것이다. 이를 Revised bathtub curve라고 부르는 것 같다.



3. Discuss about failure propagation between hardware, software and operator

⇨ Hardware failure는 초기에 많은 failure와 error들을 가진다. Software는 주로 Ideal한 hardware를 가정하여 개발을 하기 때문에 이러한 많은 error는 Software failure또한 증가시킬 수밖에 없다. 또한 hardware lifetime이 지나 생겨나는 많은 failure또한 software failure를 불러올 것이다.

Software에서의 초기 Error들이 Hardware Error을 불러올 순 있지만 Hardware Error가 Software Error에게 미치는 영향만큼 직접적인 영향을 줄 것이라 생각하긴 어려워 보인다. 하지만 잘못된 사용에 의한 Human Error 로 야기되는 Hardware Error, Software의 비 정상적인 자원 남용으로 인해 Hardware의 Lifetime의 감소 등을 생각할 수 있을 것 같다.

4. Discuss and remember about the system engineering process and the relationships between conceptual design, procurement, development, and operation.

⇨ Conceptual design은 system개발 초기에 system의 목적, 및 Client의 요구사항을 명확히 하는 것을 의미하며, 이에 따라 Procurement과정 및 이후에 필요한 Development이 이루어지게 된다. Conceptual design이 명확하게 되지 않는다면 이후의 과정들이 의미가 없어질 수도 있다.

Procurement은 High-level에서 system의 requirement를 정의하며, system component들의 분산 및 구매 결정이 이루어지게 된다. 구매가 가능한 경우엔 구매하며, 그렇지 못한 경우 Development를 진행하게 된다.

Development에 의해 필요한 system이 개발되며, operation과정은 개발 이후의 system의 배포와 관리과정이다.

위 단계들은 stage by stage로 나아가기만 하는 것이 아니며, 계속하여 넘나들 수 있다. Procurement, development 극 초기과정에서 Conceptual design을 수정해야 하는 경우도 생기며, Development과정에서 Procurement를 수정하거나, 추가로 구매결정을 하는 경우, Operation과정에서 유지 보수 혹은 Update, Upgrade를 위해 development, procurement단계로 회귀할 수도 있다.

5. What do you think is necessary for parallel development of large systems?

⇨ Parallel development는 System의 subsystem들이 병렬적으로 개발 혹은 구매되어 이후에 점진적으로 합쳐 짐으로 무엇보다 Plan-driven approach가 중요하여, Conceptual design, Procurement과정이 굉장히 중요하다고 할 수 있을 것 같다. 또한 Engineer간의 소통은 늘 강조해도 부족하다.

6. Discuss and remember the system development process from requirement development to system deployment.

⇨ Requirement Engineering

- Conceptual design 과정, Business requirement, High-level design을 분석, 문서화 하는 과정. 추상적인 관점의 Functional requirement와 Non-functional requirement, system properties를 정의하는 것이 중요하며, Undesirable characteristics를 정의하는 것도 좋다.

Architectural design

- 전반적인 Architectural와 구성요소들, 그리고 요소간 관계를 정립하는 과정. System Design과 Requirement의 feedback은 초기단계에서 필수적이고, 중요하다. Requirement and design spiral을 통해 System Requirement, Design이 문서화 된다.

Requirements partitioning

- 앞서 정한 Requirement들을 Partitioning하여 구분하여, 모아서 Sub-system을 결정, 및 sub-system의 functional한 역할을 명세화 한다. 또한 Sub-system간의 Interface, Interaction을 약속하는 과정이 포함된다.

Subsystem engineering

- 각 Subsystem을 개발, 구매하는 과정, Hardware, software 개발, operational process정의 및 Re-designing business process가 포함된다. 기본적으로 Parallel Project으로 진행되며 COST혹은 Development중에 선택하여 진행된다.

  여러 나라에서 여러 Engineer들이 병렬적으로 개발하므로, Communication이 무엇보다 중요하다.

System integration

- 개발된 Subsystem들을 통합하여 하나의 System을 만들어내는 과정.

  Hardware, software, 또 이를 사용하는 User에 대한 전체적 통합이며, 한날 한시에 전체를 통합하는 Bigbang 방식 보다는 점진적인 통합 과정이 더 유리한데, 이는 Error가 발견될 경우 어떤 Subsystem의 문제인지 판별하기가 쉽기 때문이다.

System testing

- Testing 과정을 통해 Error, Problems를 찾아내는 과정

System deployment

- Customer의 Environment에 설치, 배포하는 과정, 대규모의 시스템은 많은 시간이 걸리기도, 해결하기 어려운 Error들을 발견하기도 한다. 또한 기존의 사용자들이 새로운 시스템을 반기는 경우는 거의 없다.

7. Discuss why procurement decisions affect system functionality, such as reliability.

⇨ Procurement 과정에서의 잘못된 Plan은 여러가지 문제를 야기할 수 있을 것으로 보인다.

Interface에 대한 불완전한 정의는 개발된 Subsystem을 합치고, 사용하는데 문제가 될 수 있으며, Architecture Design의 문제는 system에 대한 많은 Hardware Error들을 야기하고, 이는 Software Error까지 연결될 수 있다. 또한 Hardware의 Timing faults와 같은 Error들은 Software의 Reliability Problem들과도 연결된다.

또한 외부에서의 구매를 결정하는 경우, 악의를 가진 외주업체에 의해 Reliability Problem들이 야기될 수 있다. 이는 Deep - Learning에서의 Data Poisoning, Back door attack 등을 이야기할 수 있을 것 같다.

8. Discuss how the various activities of the system development process affect system dependability.

⇨ 여러 Error는 System에서 필연적으로 발생할 수밖에 없으며, 이는 System의 Dependability에 영향을 미친다. 이는 사용자에 의한 Human Error일 수도 있으며, hardware자체 결함, Expire of Hardware Lifetime, Software Update, Upgrade과정에서의 Failure, 등 일 수 있다.

9. Discuss one or more examples of the various barriers to protect human error

⇨ OS의 Journaling file system을 이야기할 수 있을 것 같다. 이는 Sudden Power off등의 상황에서 Crash를 방지하고, File System consistency를 유지시키며, File system에서의 Barrier 역할을 하고 있다.

또한 Network System의 TCP Loss packet Recovery과정 또한 System의 Barrier 역할을 한다고 이야기할 수 있다.

Hardware level의 Error Correcting Code 또한 System의 Barrier라고 부를 수 있을 것 같다.

10. Discuss and summary about why the evolution is costly.

⇨ Evolution은 많은 요인들에 의해 이루어질 수 있다. 고객의 요구사항 일수도, 기술환경이 바뀔 수도 있으며, 새로운 표준, 경쟁 업체의 새로운 제품 등 여러 문제에 따라 Evolution이 진행된다. 이는 Evolution이 기본적으로 새로운 변화사항에 대한 Technical, Business적인 검토가 필요하다. 또한 새로운 유지, 보수가 필요한 이유에 대한 충분한 근거와 자료가 필요하여, Evolution은 기본적으로 Highly cost를 가진다.

또한 거대한 system은 subsystem간의 interaction을 모두 파악하기가 어려우며, 이는 유지, 보수에 의해 예측하지 못한 오류를 야기할 수도 있기 때문에 Evolution은 costly 하다.

Questions of yourself

1. Stakeholders들의 특성이 Conceptual design 과정에 어떤 영향을 미칠 수 있는가?

⇨ Stakeholders들은 다양한 view를 가지고 있기 때문에 Stakeholder의 스펙트럼이 넓어질 수록 각각의 요구사항과, 지향점을 맞추기가 어려워진다 따라서 Stakeholder의 스펙트럼은 Conceptual Design과정을 굉장히 어렵게 만들 수 있다.

2. 새로운 System은 Organization에 어떤 영향을 미칠 수 있는가?

⇨ Process Changes

- System은 Organization에서 사용되기 위하여 Training과정이 필요하다.

Job Changes

- System은 User들의 기술 혹은 작업방식에 영향을 미칠 수 있다.

Organizational Changes

- System은 Organization의 권력계층 혹은 이해관계에 영향을 미칠 수 있다.