SOFTWARE MODELING & ANALYSIS
OOAD VS. SASD

200412338 이선희 200511309 김의섭

목차

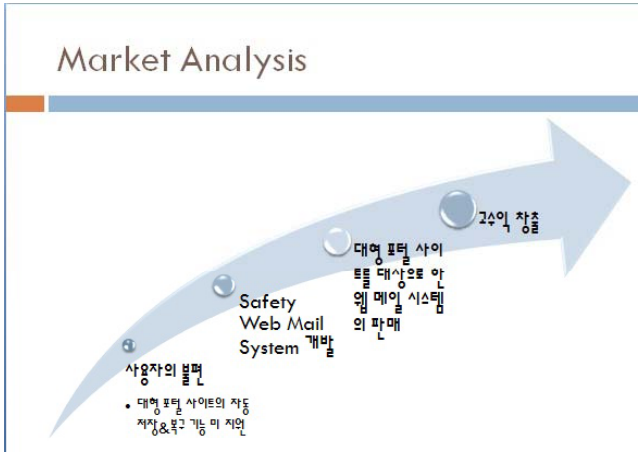
- OOAD
  - OOAD의 특징
  - OOAD의 장점
  - OOAD의 단점
- SASD
  - SASD의 특징
  - SASD의 장점
  - SASD의 단점
- OOAD vs. SASD
  - 특징의 비교
  - 우리는 어떠한 기준으로 개발 방법론을 택하여야 하는가

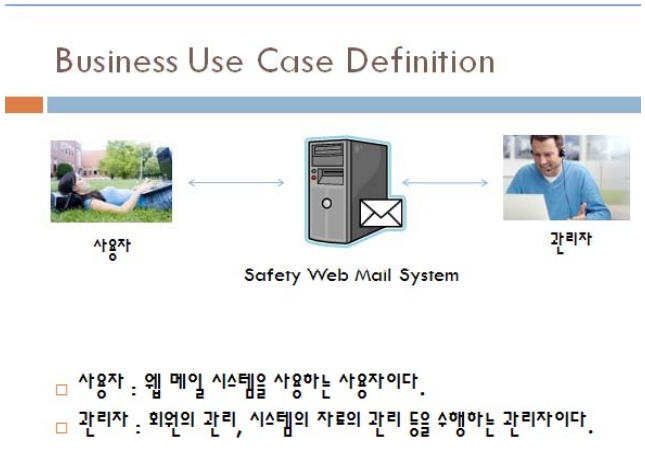
OOAD

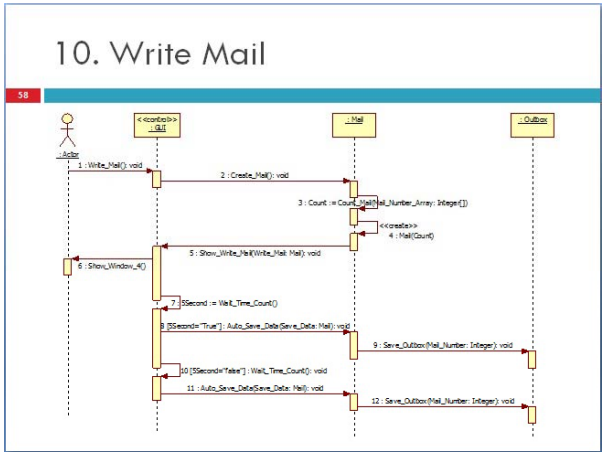
- OOAD의 특징
- OOAD의 장점
- OOAD의 단점

OOAD
(Object Oriented Analysis & Design

- Object-Oriented
  - 객체를 지향하는 설계 : 크게 Use Case, Class Diagram, Sequence Diagram으로 표현 된다.
- SASD보다 늦게 나온 SE Process이다.
  - SASD보다 더욱더 디테일한 명시가 가능하다.
  - 사업자에게 제시할 프로젝트 제안서를 제작하는 과정이 포함되어 있다.







OOAD

2041. Design Real Use Cases

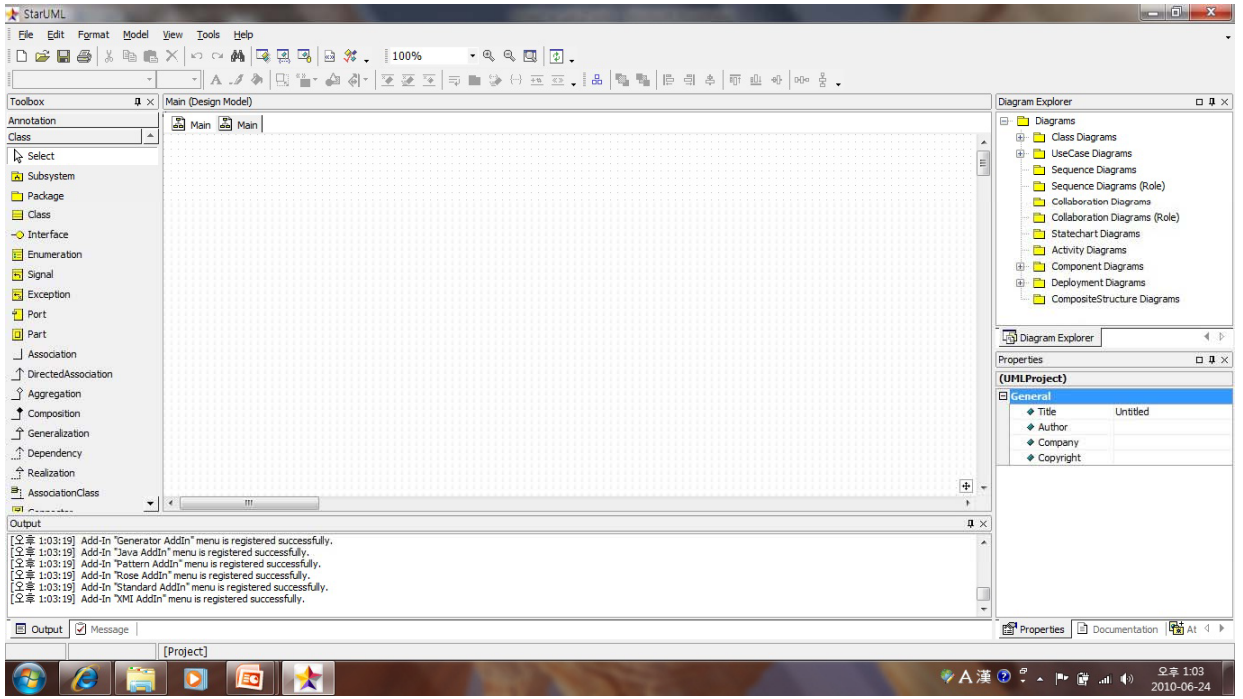
21

Use Case	12. Send Mail
Actor	사용자
Purpose	사용자가 작성한 메일을 보낸다.
Overview	작성된 메일을 발송하는 '메일 보내기'이다. 이 기능이 수행되면 매당 메일은 자동적으로 '보낸 메일 함'에 저장 된다
Type	Primary & Real
Cross Reference	System function : R2.2, R3.1, R3.3, R3.4 Use Case : Sent Mail Box, Write Mail, Save Mail to Send Mail Box
Pre-Requisites	사용자는 메일 작성 중이어야 한다.
UI Widgets	Window-4
Typical Courses of Events	(A) : Actor , (S) : System 1. (A) 보내기를 클릭한다. 2. (S) ID중에 Receive_member를 검색한다. 3. (S) 검색된 ID의 Mail 객체를 생성한다. 4. (S) Receive_member의 Received_Mail_Box의 Mail_List에 생성된 Mail객체의 Mail_Number를 추가한다. 5. (S) Save Mail to Sent Box를 실행한다.
Alternative Courses of Events	None
Exceptional Courses of Events	3. '받는 사람'에 적합 주소가 없을 경우 메일 작성 페이지로 넘어가고 재입력을 받는다.

↑ 상세한 명세를 요구하는 OOAD

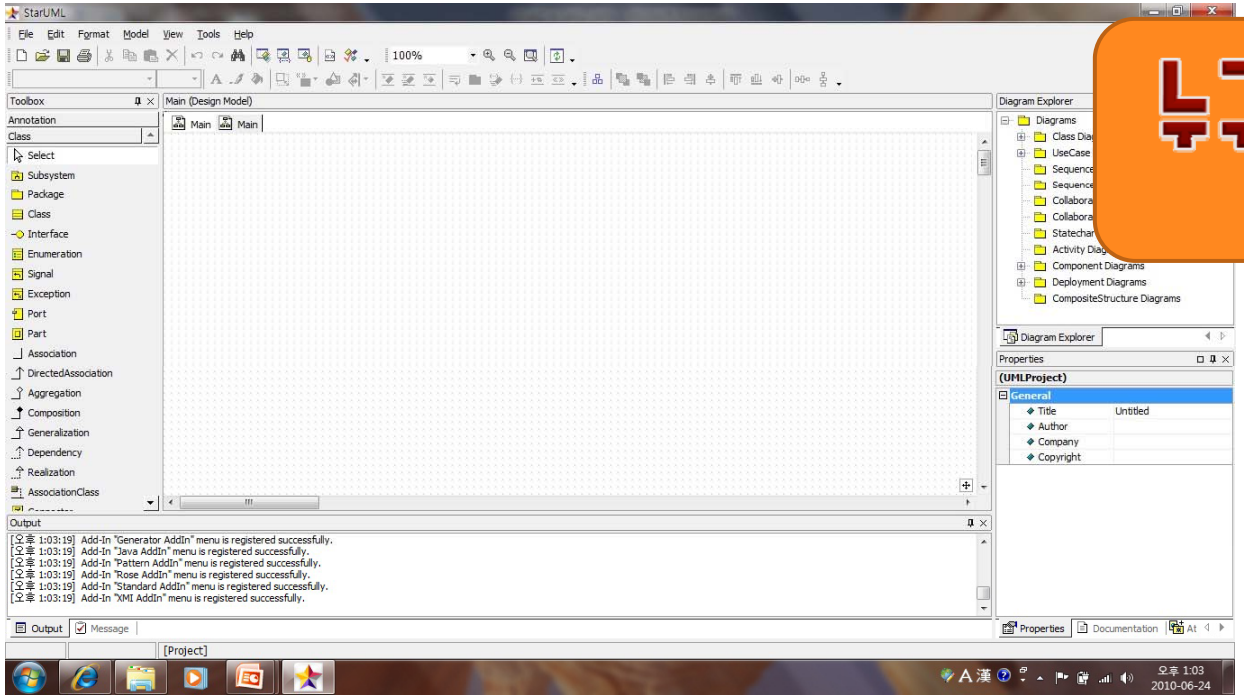
- 각각의 Use Case별로 상세한 명세를 요구한다.
  - 각각의 Use Case마다의 메인 시나리오는 흐름을 알 수 있게 한다.
  - Use Case사이의 관계를 알 수 있다.
  - 예외사항의 처리 또한 쉽게 알 수 있고, 추가, 삭제할 수 있다.

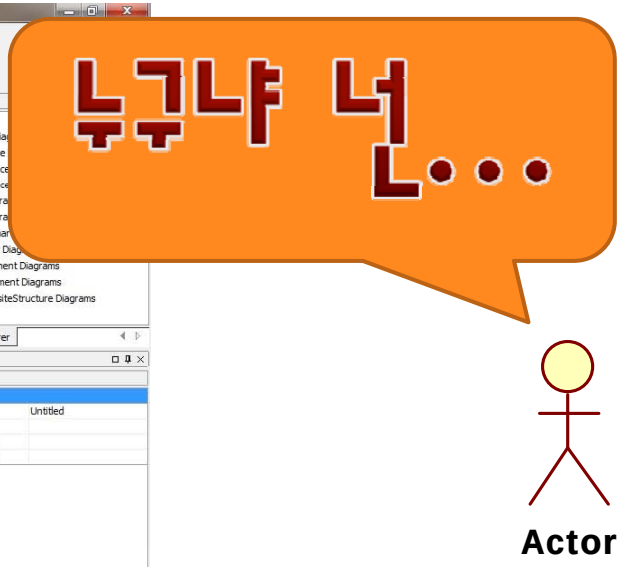
OOAD



- 개발 tool이 있다.
  - 보다 손쉽게 Diagram들을 제작할 수 있는 개발 툴은 반복 작업시의 수정, 추가, 삭제 등을 보다 간편하게 할 수 있게 도와준다.

OOAD





- UML tool은 처음 접하는 사람에게는 힘들다. 새로운 tool의 존재로 인해, 작업의 편의성은 늘어난다고 할 수는 있겠으나, 또 다른 도구 사용법을 배워야 한다는 점은 부담감으로 작용할 수 밖에 없다.

OOAD

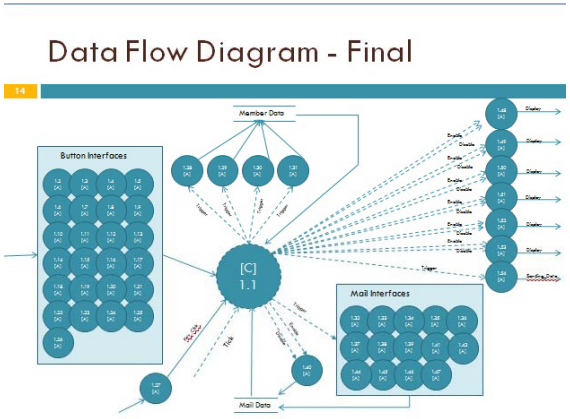
- 객체를 지향하는 보다 큰 규모의 프로젝트를 위한 방법론이다.
  - 개발 팀원들 간의 대화를 위해 Use Case에 대해서도 각 단계마다 반복적인 재 정의를 요구한다.
  - 불필요한 문서 작성의 시간이 개발 시간보다 많은 시간을 차지할 수 있다.
  - 두 명 같은 적은 인원이 하려 했다간 문서 더미에 갇혀 숨을 못 쉬는 사태가 발생 가능하다.

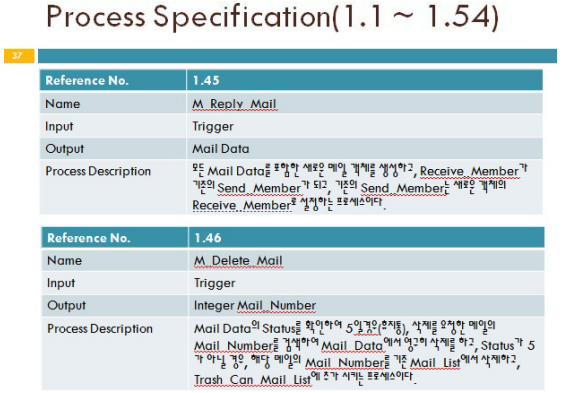
SASD

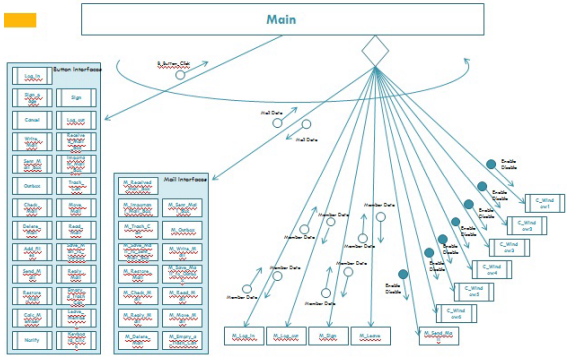
- SASD의 특징
- SASD의 장점
- SASD의 단점

SASD
(Structured Analysis & Structured Design)

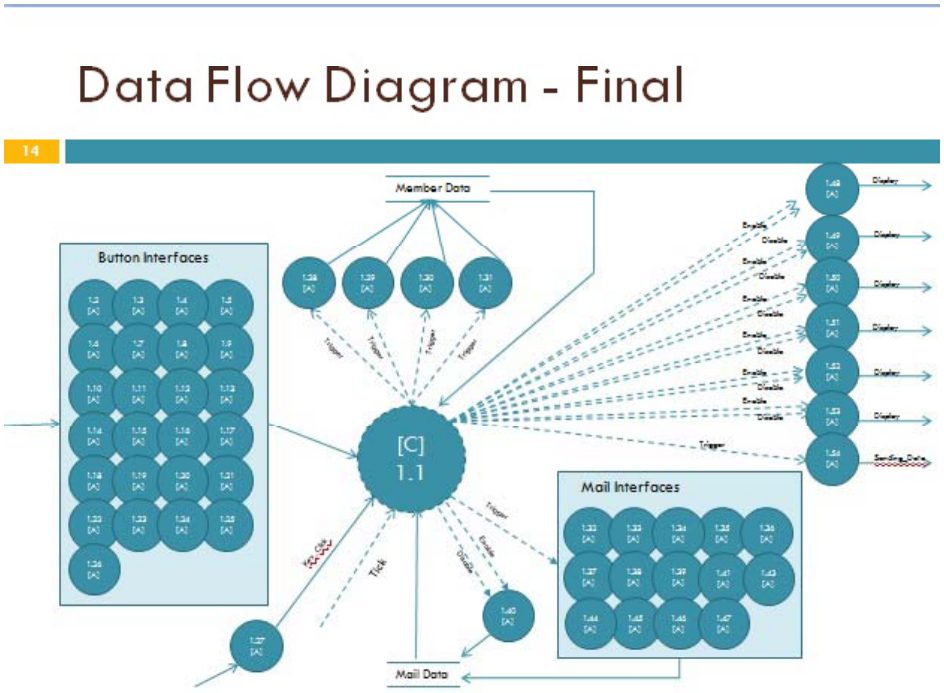
- Top-Down 방식으로 프로그램을 구성한다.
  - System Context Diagram에서 출발하여 세부적인 사항들을 차례차례 해결해 나간다.
- Data의 흐름을 통한 프로그램의 구조를 설계하는 방법론이다.
  - DFD를 통해 우리는 Data의 흐름을 알 수 있다.
  - 프로그램의 전체적인 구조를 알 수 있으며, Structure Chart를 통해 바로 프로그램을 구성할 수 있다.







SASD



- Data Flow Diagram은 시스템의 전체 데이터 흐름을 한눈에 알아볼 수 있도록 도와준다.
- Input과 Output이 명확하게 구성되어 있어, Embedded System에는 매우 적합하다 할 수 있다.

SASD

Process Specification(1.1 ~ 1.54)

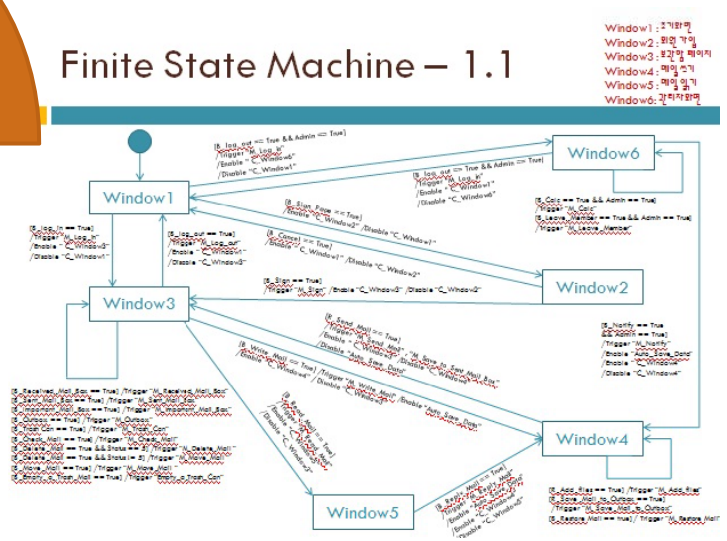
37

Reference No.	1.45
Name	M_Reply_Mail
Input	Trigger
Output	Mail Data
Process Description	모든 Mail Data를 포함한 새로운 메일 객체를 생성하고, Receive_Member가 기존의 Send_Member가 되고, 기존의 Send_Member는 새로운 객체의 Receive_Member로 설정하는 프로세스이다.

Reference No.	1.46
Name	M_Delete_Mail
Input	Trigger
Output	Integer Mail_Number
Process Description	Mail Data의 Status를 확인하여 5일경우(휴지통), 삭제를 요청한 메일의 Mail_Number를 검색하여 Mail_Data에서 영구히 삭제를 하고, Status가 5가 아닐 경우, 해당 메일의 Mail_Number를 기존 Mail_List에서 삭제하고, Trash_Can_Mail_List에 추가 시키는 프로세스이다.

- Process Specification은 너무나도 적은 정보를 포함한다.
  - 적은 정보를 포함하는 이유는, 하나의 프로세스가 최소한의 작업을 수행하는 것이라는 개념을 위한 것이다.
  - 하지만 OOAD의 Use Case에 비해서 너무나도 적은 정보를 가지기 때문에 의미의 전달이 왜곡되거나, 되지 않을 수 있다.

SASD



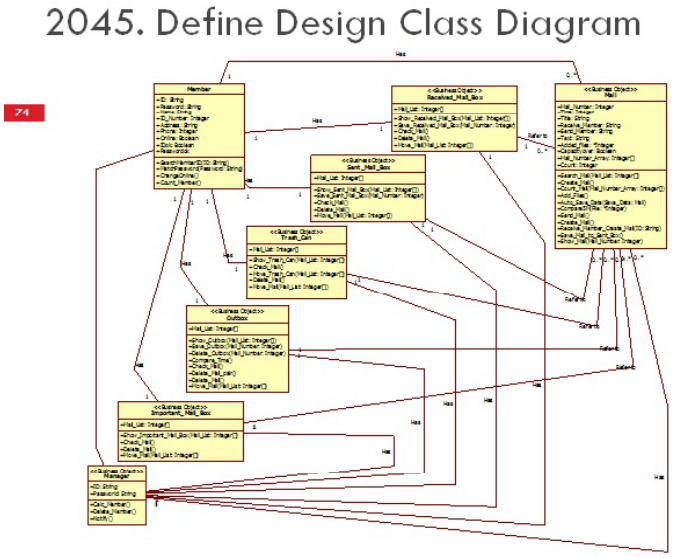
- 세부적인 조건에 따른 시스템 상태의 변화가 지나치게 Finite State Machine에 의존적이다.
- 시스템이 더 복잡해질 수록 더 복잡한 Finite State Machine이 생겨날 것이며, 이것의 개수 또한 늘어날 것이다.

OOAD vs. SASD

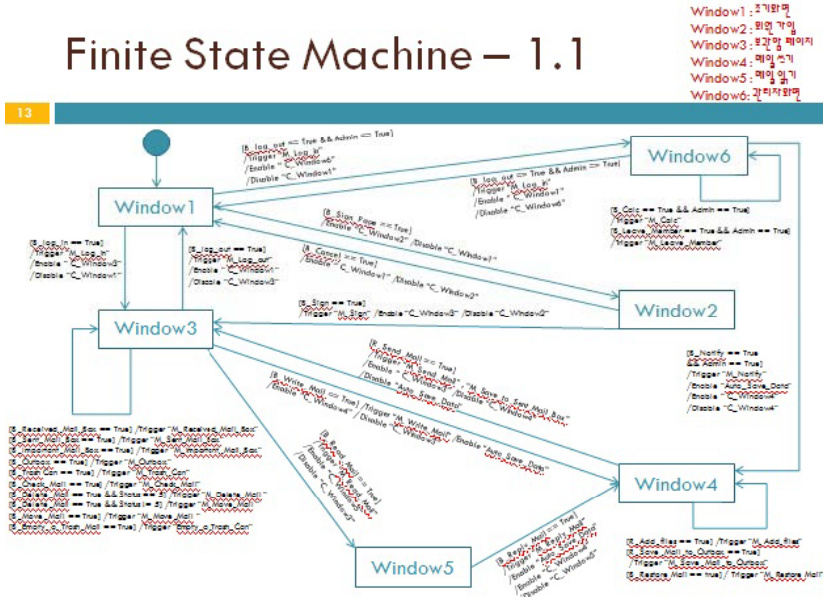
- 특징의 비교
- 우리는 어떠한 기준으로 방법론을 택하여야 하는가

OOAD vs. SASD

- 두 방법의 특징 비교
  - 프로그램 내의 관계와 상태
    - OOAD : Class들의 관계를 보다 명확하게 알 수 있다.
    - SASD : Finite State Machine을 통해 시스템의 상태 변화를 알 수 있다.



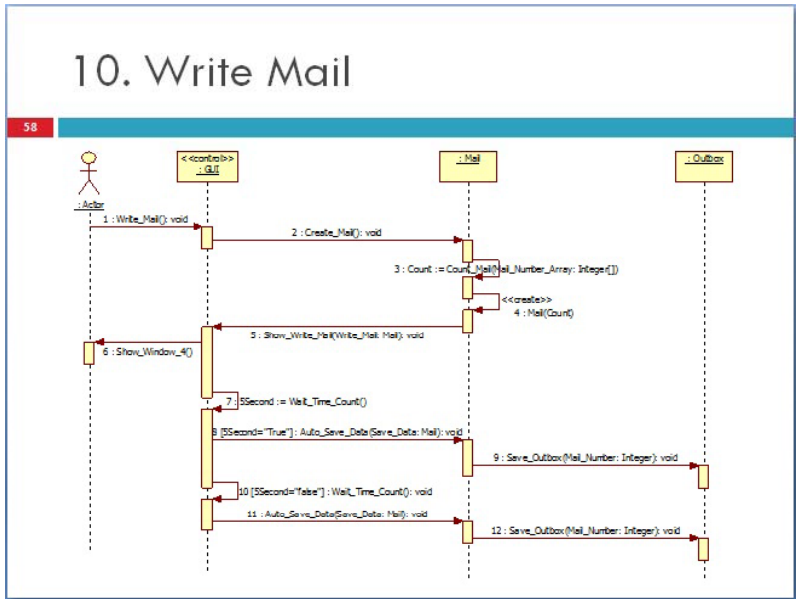
Class Diagram



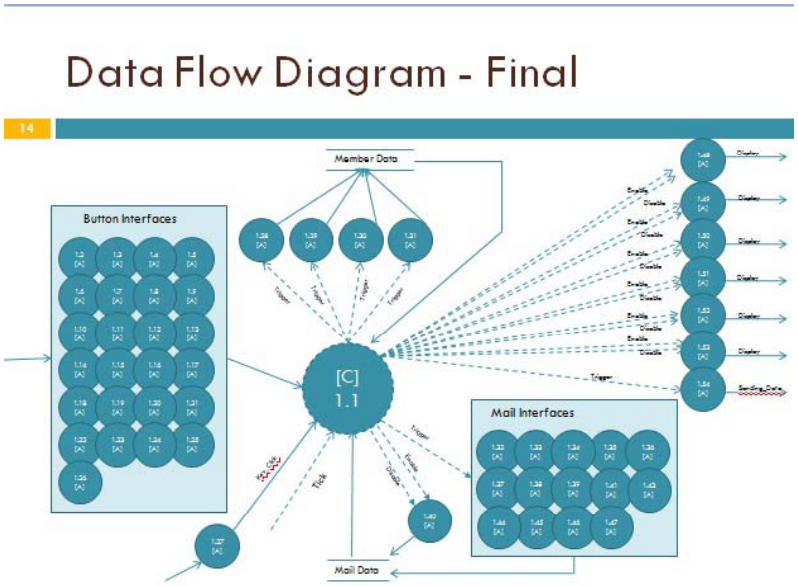
Finite State Machine

OOAD vs. SASD

- 데이터의 흐름, 함수의 흐름
  - OOAD : Sequence Diagram을 통해 함수가 호출되는 순서를 명확하게 알 수 있다.
  - SASD : Data Flow Diagram을 통해 데이터가 어떻게 전달되는지를 알 수 있다.



Sequence Diagram



Data Flow Diagram

OOAD vs. SASD

- 프로그램을 개발할 때, 우리는 어떠한 기준으로 개발 방법론을 택하여야 하는가.
  - 분할 작업의 가능성
    - 여러 명의 인원이 동시에 작업을 하는 경우라면 객체 지향 적인 OOAD를 선택하는 것이 바람직하다.
  - 요구사항의 빈번한 변화
    - 사용자의 요구사항이 지속적으로 변화한다면, 보다 수정이 용이한 OOAD를 선택하는 것이 바람직하다.
  - 작업의 규모
    - 개발하고자 하는 프로그램의 규모가 작다면, SASD를 선택하는 것이 작업의 효율에서 월등히 앞선다.