MVC 패턴

Model
View
Control Pattern

아꿈사 http://cafe.naver.com/architect1
TTF http://www.npteam.net

상호 작용 시스템

- 구성요소
  - 핵심 기능
  - 사용자 인터페이스

- 핵심 기능
  - 시스템의 기능적 요구사항에 기초한다.
  - 대체로 변경 없이 유지된다.

- 사용자 인터페이스
  - 변경되기 쉽다.
  - 상황에 따라 유연하게 적용해야 한다.

상호 작용 시스템의 2가지 패턴

Model
View
Controller

Presentation
Abstraction
Control

각 패턴의 사용 사례

MVC 패턴

- MFC 라이브러리
- 스몰토크

PAC 패턴

- AI(artifical intelligence)

MVC 패턴의 3가지 컴포넌트

Model Component
- 핵심기능
- 데이터

View Component
- 사용자에게 정보를 출력

Controller Component
- 사용자 입력을 처리

MVC Pattern - Context

- 정황(Context)
  - 상호작용 Application에 유연한 사람-컴퓨터간 인터페이스를 구축해야 한다.

매튜 네이글은 뇌에 이식된 칩을 컴퓨터에 연결하여 생각으로 로봇팔, 마우스 포인터를 움직인다.

이미지 출처 : http://www.scienceahead.com/entry/rsearchers-to-develop-a-brain-control-interface-device-that-could-be-manipulated-by-heartbeat/

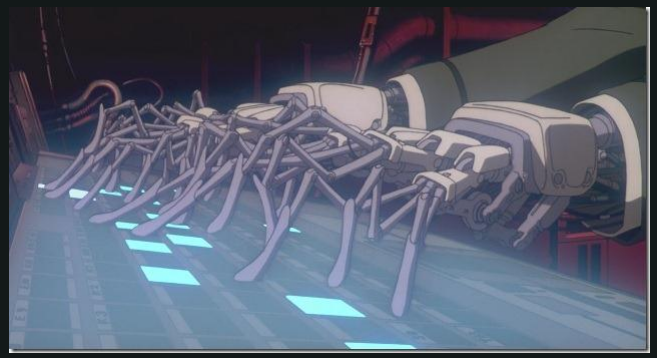
MVC Pattern - Problem

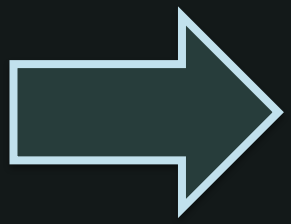
- 문제(Problem)
  - 고객의 요구에 따라서 사용자 인터페이스는 변경될 가능성이 높다.

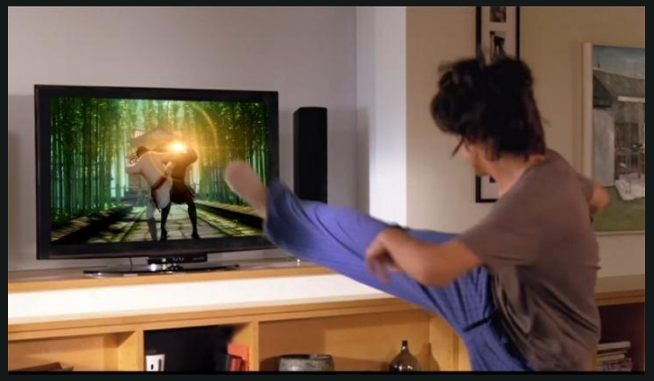
이미지 출처 : http://kr.engadget.com/2009/07/01/Victorinox-Swiss-Army/

MVC Pattern - Problem

- 문제(Problem)
  - 사용자마다 상충되는 요구사항을 통합할 수 있어야 한다.

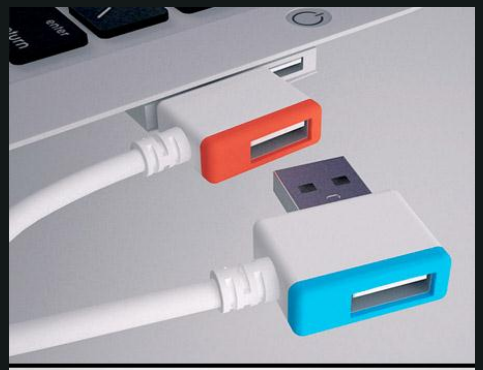






MVC Pattern - Problem

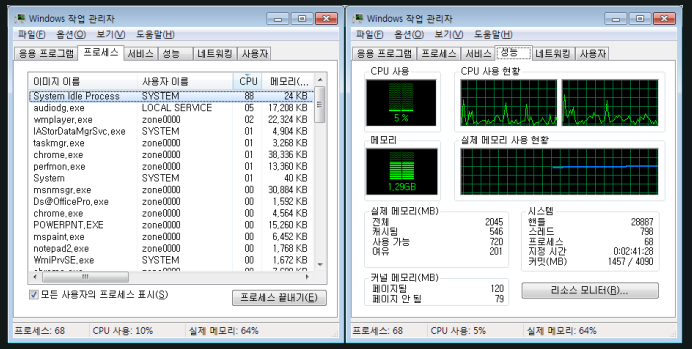
- 유연성(Flexibility)
  - 사용자 인터페이스를 분리해야 하는 이유
    ① 코드 수정 비용 증가
    ② 오류 발생 확률 증가
    ③ 결합도가 높은 모듈에 변경사항 발생시 모든 모듈을 일일이 변경해야 한다.



이미지 출처 : http://hightouch.kr/522

MVC Pattern - Problem

- 문제에 내포된 영향력(Force)
  - 동일한 정보가 서로 다른 모습으로 표시된다.



  - 데이터 변경 사항이 Application과 Display에 즉각 반영되어야 한다.

MVC Pattern - Problem

- 문제에 내포된 영향력(Force)
  - 사용자 인터페이스는 쉽게 변경될 수 있어야 한다.

  - 사용자 인터페이스 변경시 핵심코드는 어떠한 영향도 미쳐선 안 된다.

MVC Pattern – Solution

- 해법(Solution)
  - MVC는 상호작용 Application을 3가지 영역으로 분리한다.
    ① 프로세싱(processing)
    ② 출력(output)
    ③ 입력(input)

  - 3가지 컴포넌트로 구성된다.
    ① 모델 컴포넌트(**M**odel component)
    ② 뷰 컴포넌트(**V**iew component)
    ③ 컨트롤러 컴포넌트(**C**ontroller component)

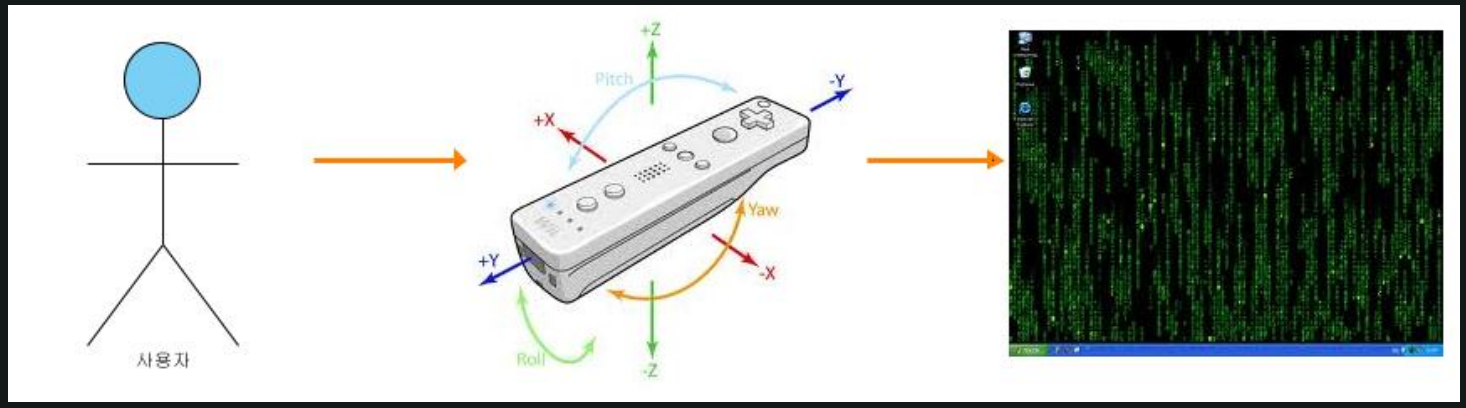
MVC Pattern – Solution

- 모델 컴포넌트(Model component)
  - 핵심 기능과 데이터를 캡슐화하여 입출력에 영향을 받지 않고 독립적이다.

- 뷰 컴포넌트(View component)
  - 사용자에게 정보를 디스플레이 한다.
  - 모델로부터 제공된 데이터를 다양한 View를 통해 표시한다.

MVC Pattern – Solution

- 뷰 컴포넌트(View component)
  - 각 View마다 컨트롤러가 하나씩 연결된다.
  - 컨트롤러는 사용자 입력을 이벤트로 받는다.
  - 사용자는 오직 컨트롤러를 통해서만 시스템과 상호작용 한다.

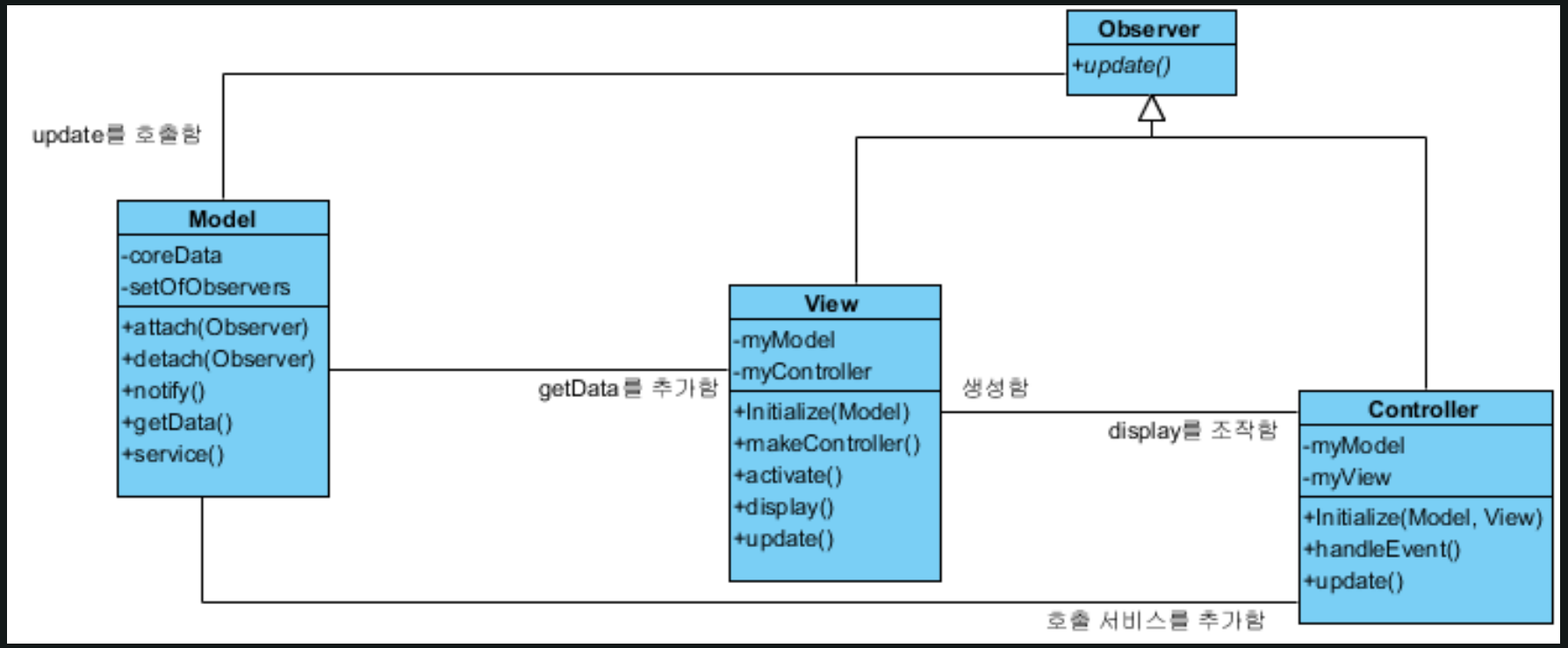


MVC Pattern – Solution

- 컨트롤러 컴포넌트(Controller component)
  - 이벤트 형태로 사용자 입력을 받는다.
  - 사용자 인터페이스 플랫폼에 따라 이벤트 전달 과정이 달라진다.
  - 모델 데이터 상태에 따라 컨트롤러 UI 변경이 가능하다.(변경 전파 메커니즘)

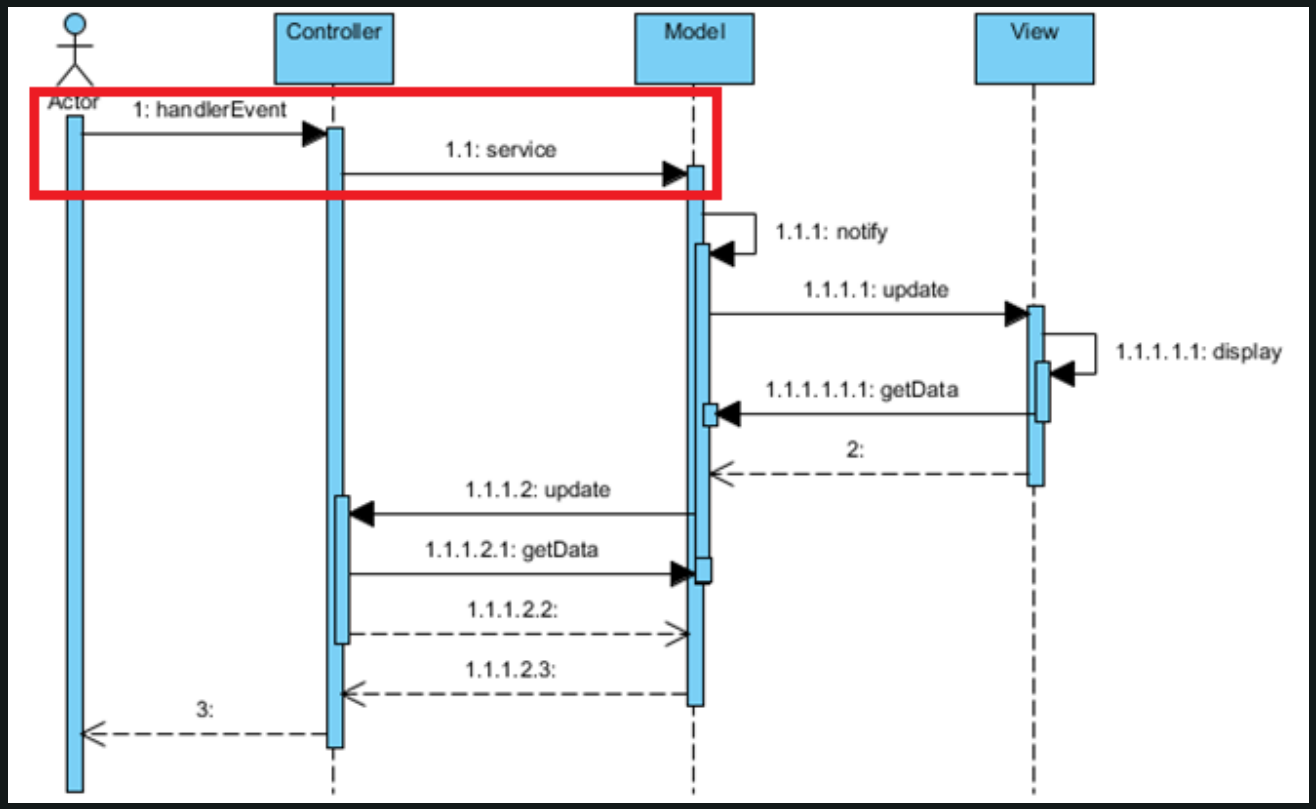
MVC Pattern – Solution

- MVC 패턴 UML 다이어그램
  - 뷰와 컨트롤러 클래스는 Update 인터페이스를 가진 기본클래스(Observer)에서 상속 받는다.



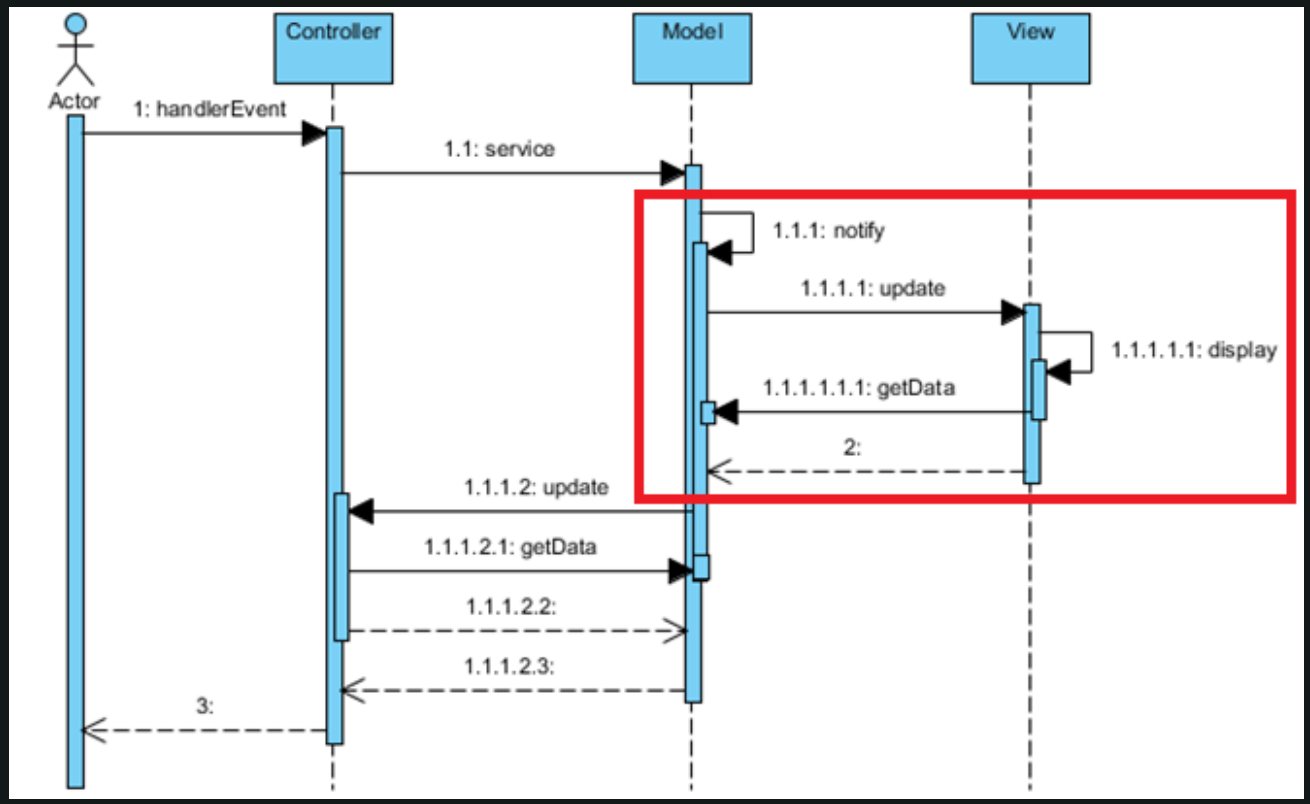
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오1)
  - 컨트롤러는 사용자로부터 이벤트를 받는다.
  - 모델의 서비스 프로시저에 이벤트를 보낸다.
  - 모델은 서비스를 수행하고 내부 데이터를 변경시킨다.



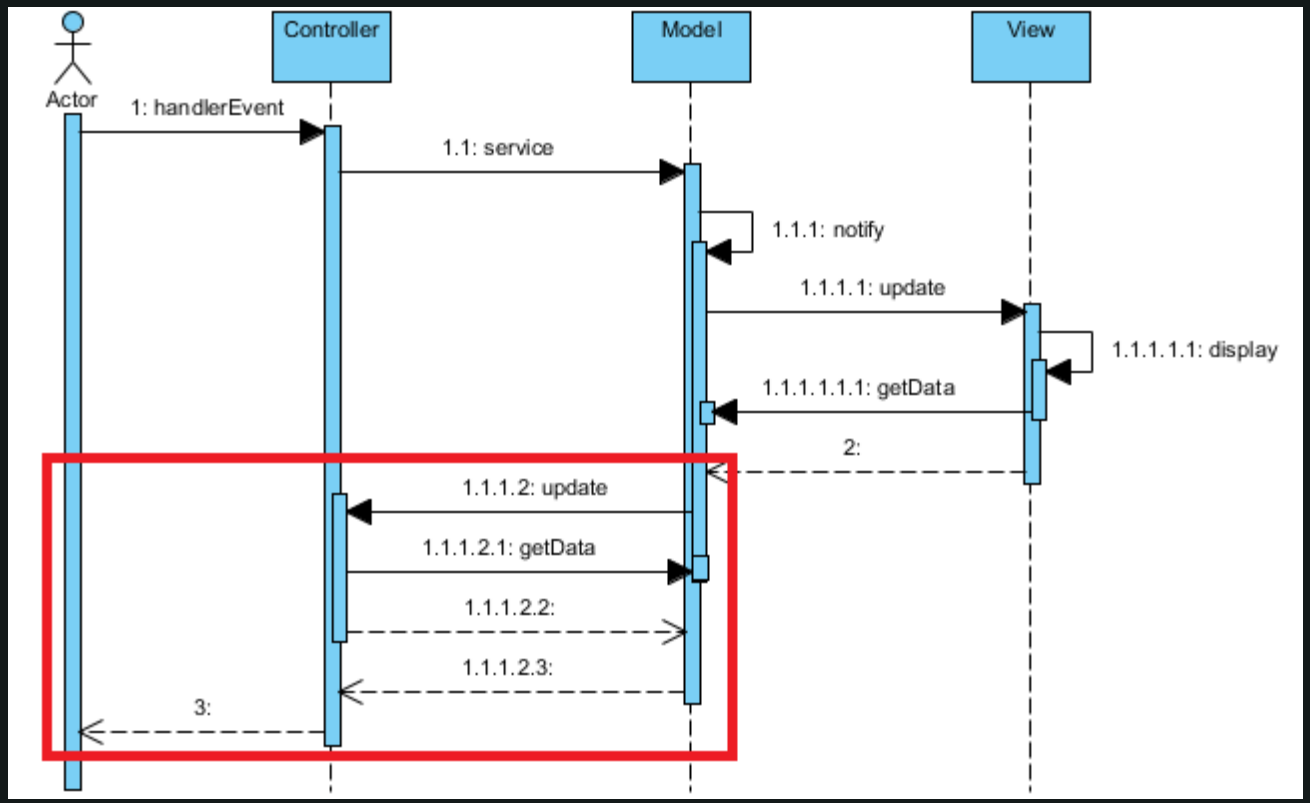
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오1)
  - 모델은 업데이트 프로시저를 호출한다.
  - 변경전파 메커니즘의 모든 뷰와 컨트롤러에 통지를 보낸다.
  - 뷰는 모델에 변경 데이터를 요청하고 화면을 갱신한다.



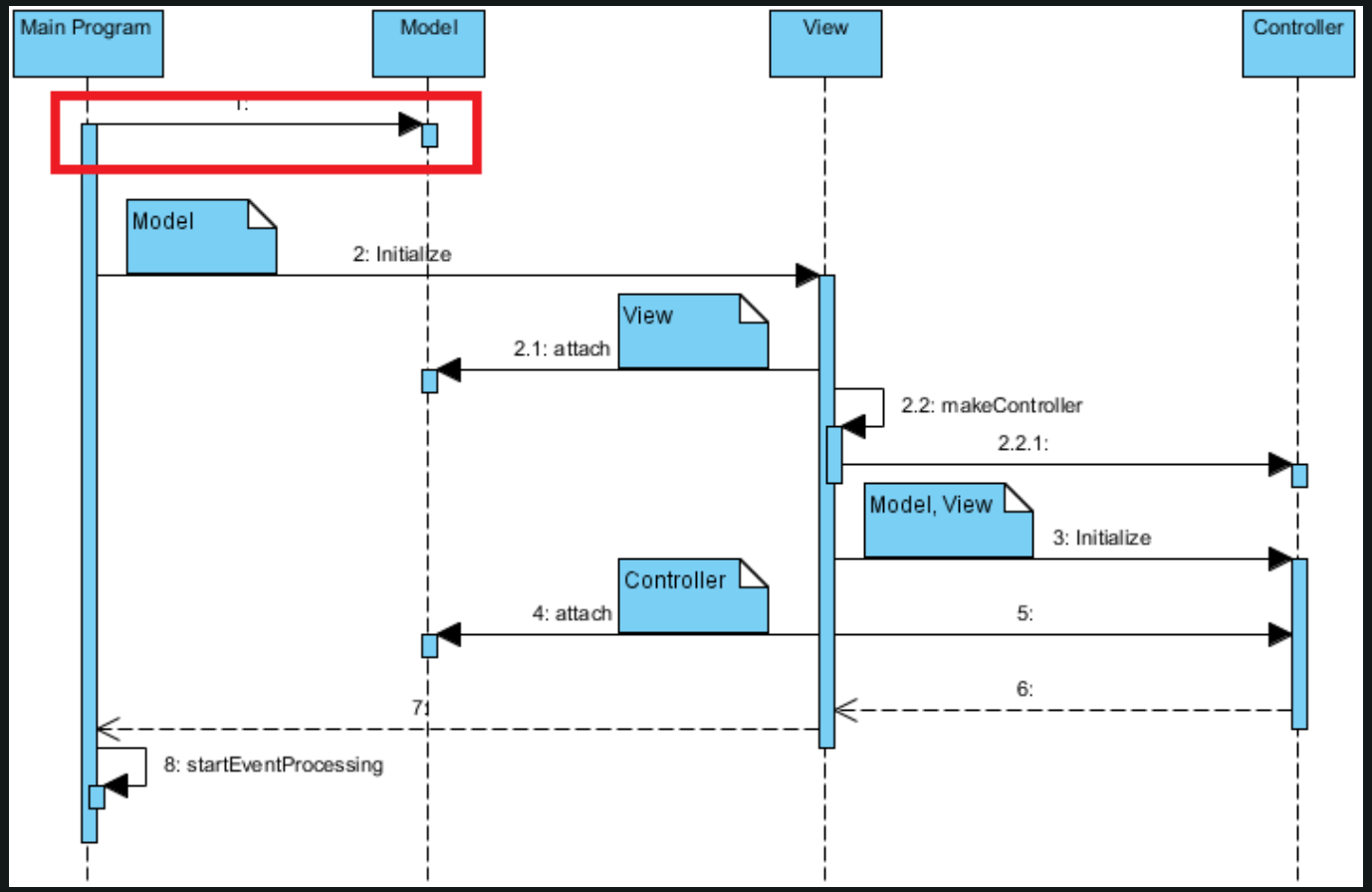
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오1)
  - 컨트롤러는 모델로부터 데이터를 가져와서 컨트롤러를 활성화/비활성화 한다.
  - 컨트롤러가 제어권을 반환 받는다.



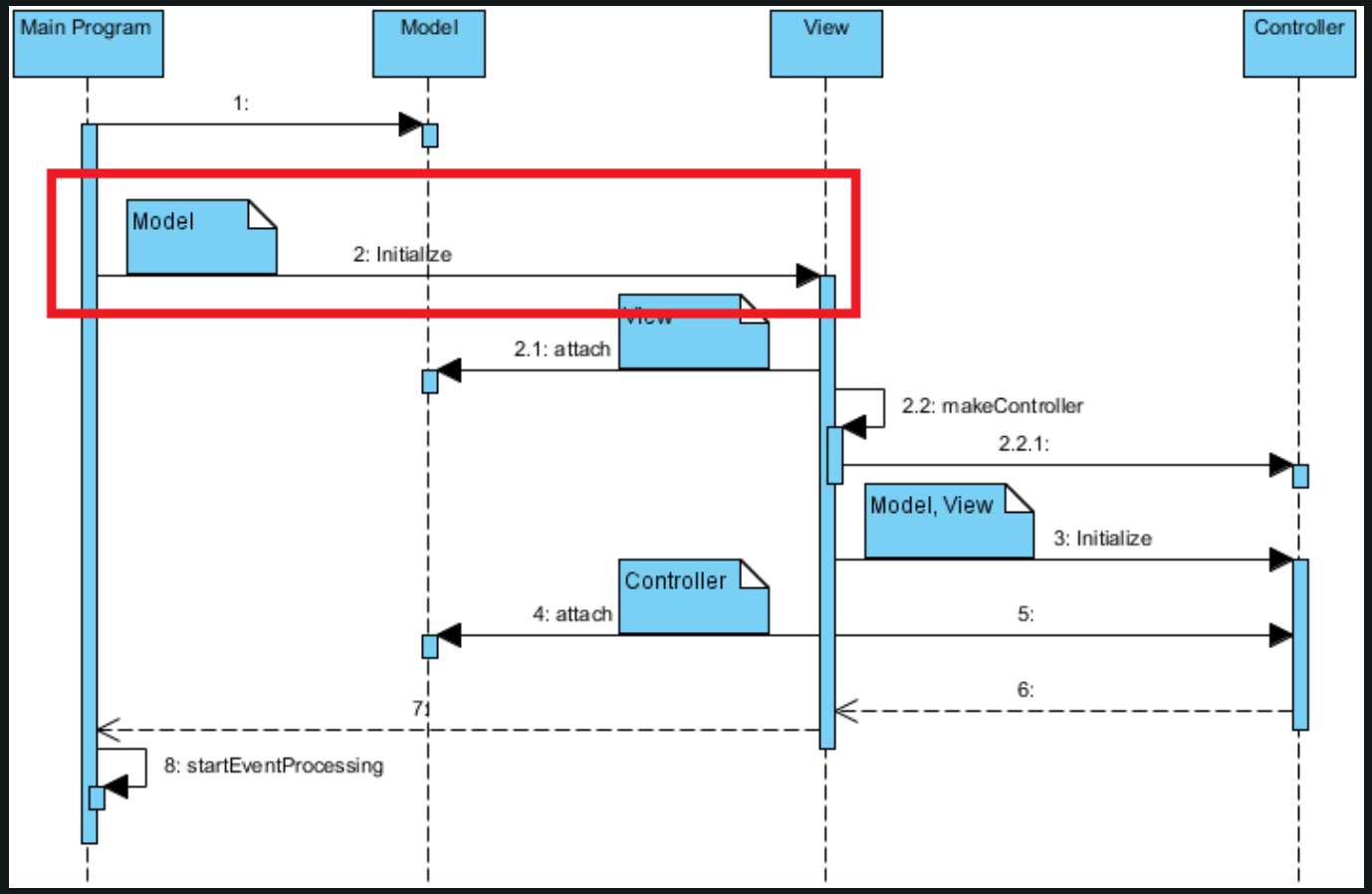
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 모델 인스턴스가 생성되며, 내부 데이터 구조 초기화



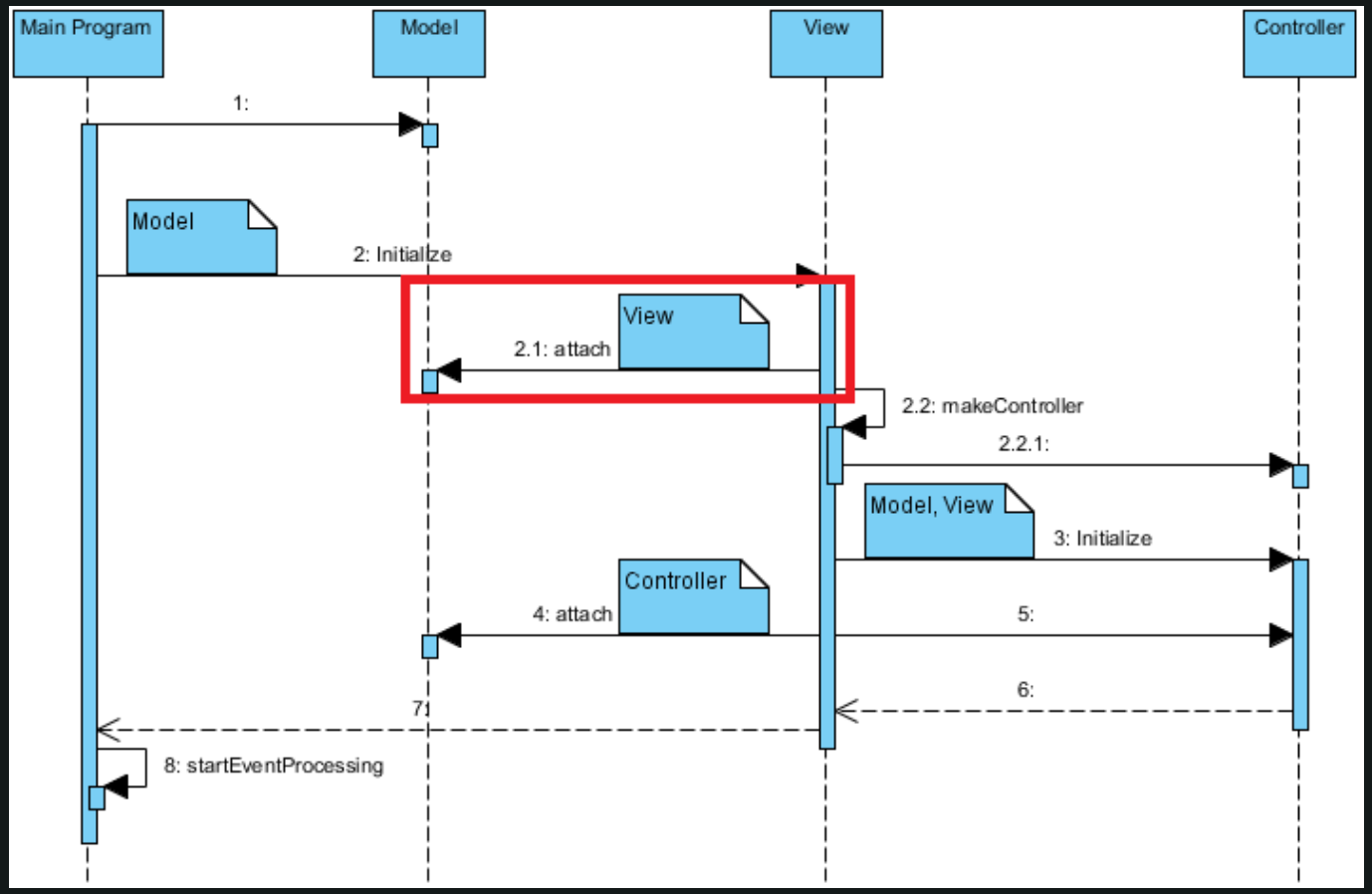
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 뷰 객체 생성.
  - 초기화를 위해 모델의 참조를 매개변수로 입력 받는다.



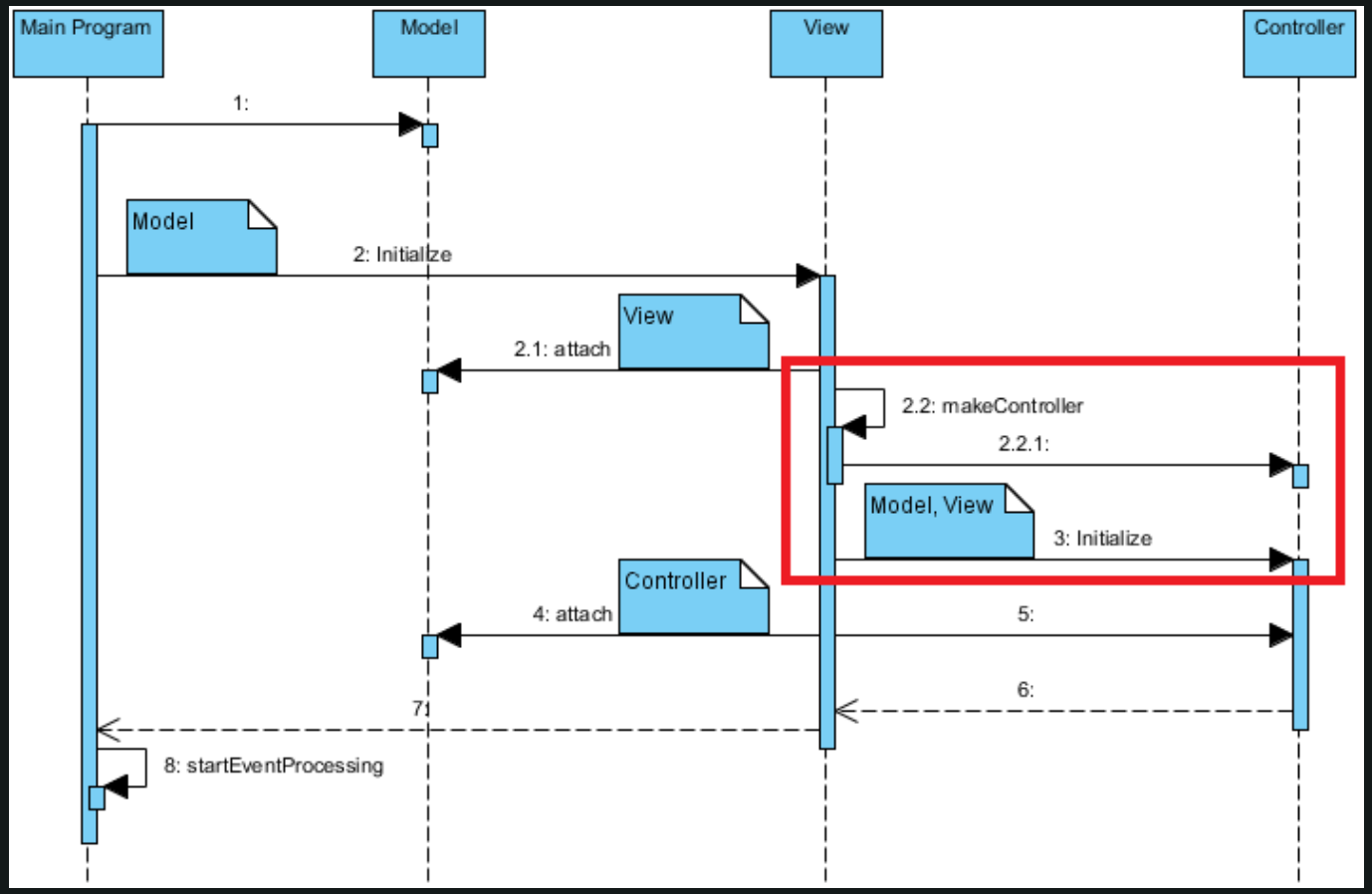
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 뷰는 attach 프로시저 호출.
  - 모델의 변경전파 매커니즘에 자신을 등록한다.



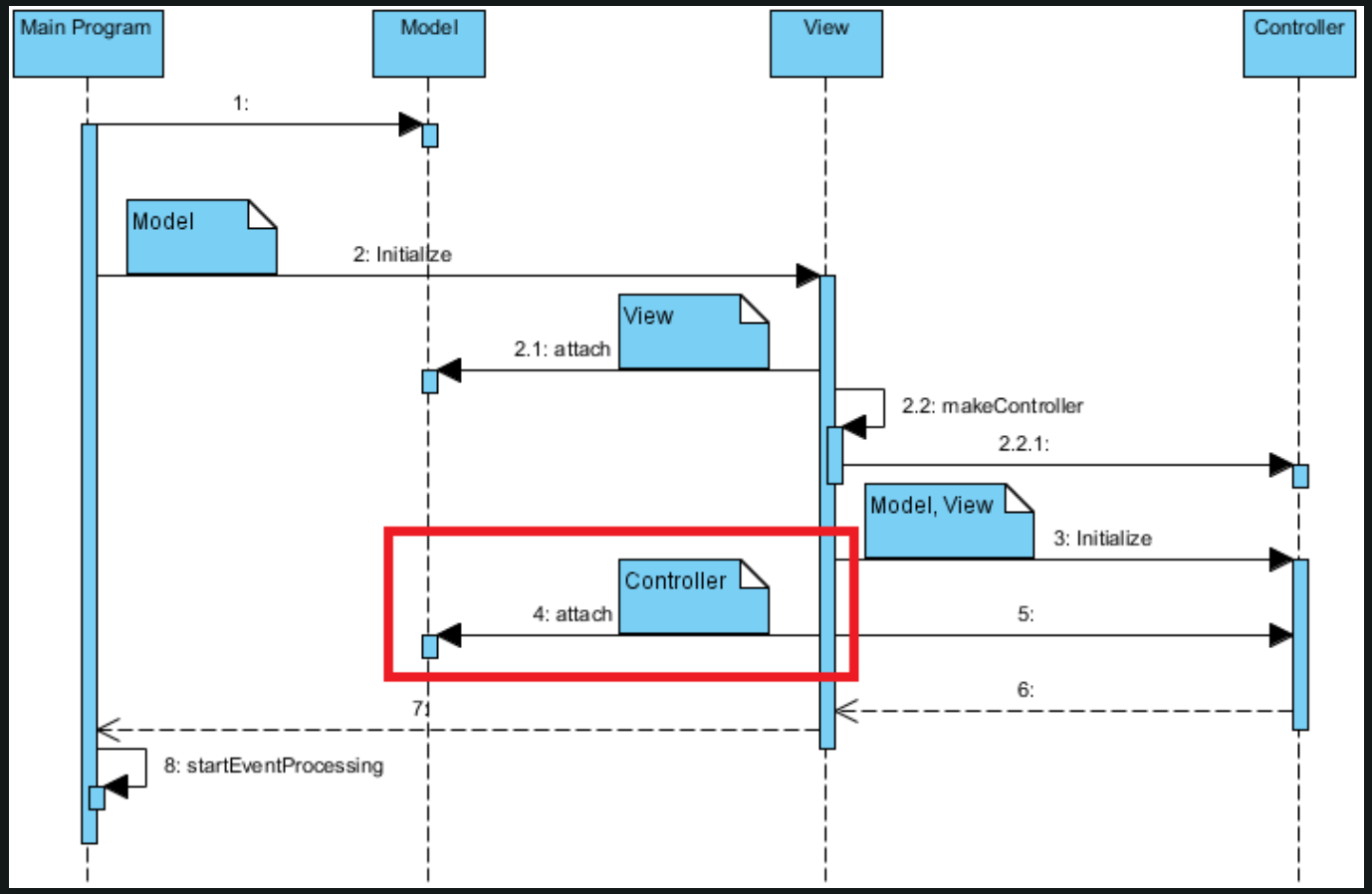
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 뷰는 컨트롤러 생성 및 초기화.
  - 뷰는 모델과 자신의 참조를 초기화 프로시저에 넘긴다.



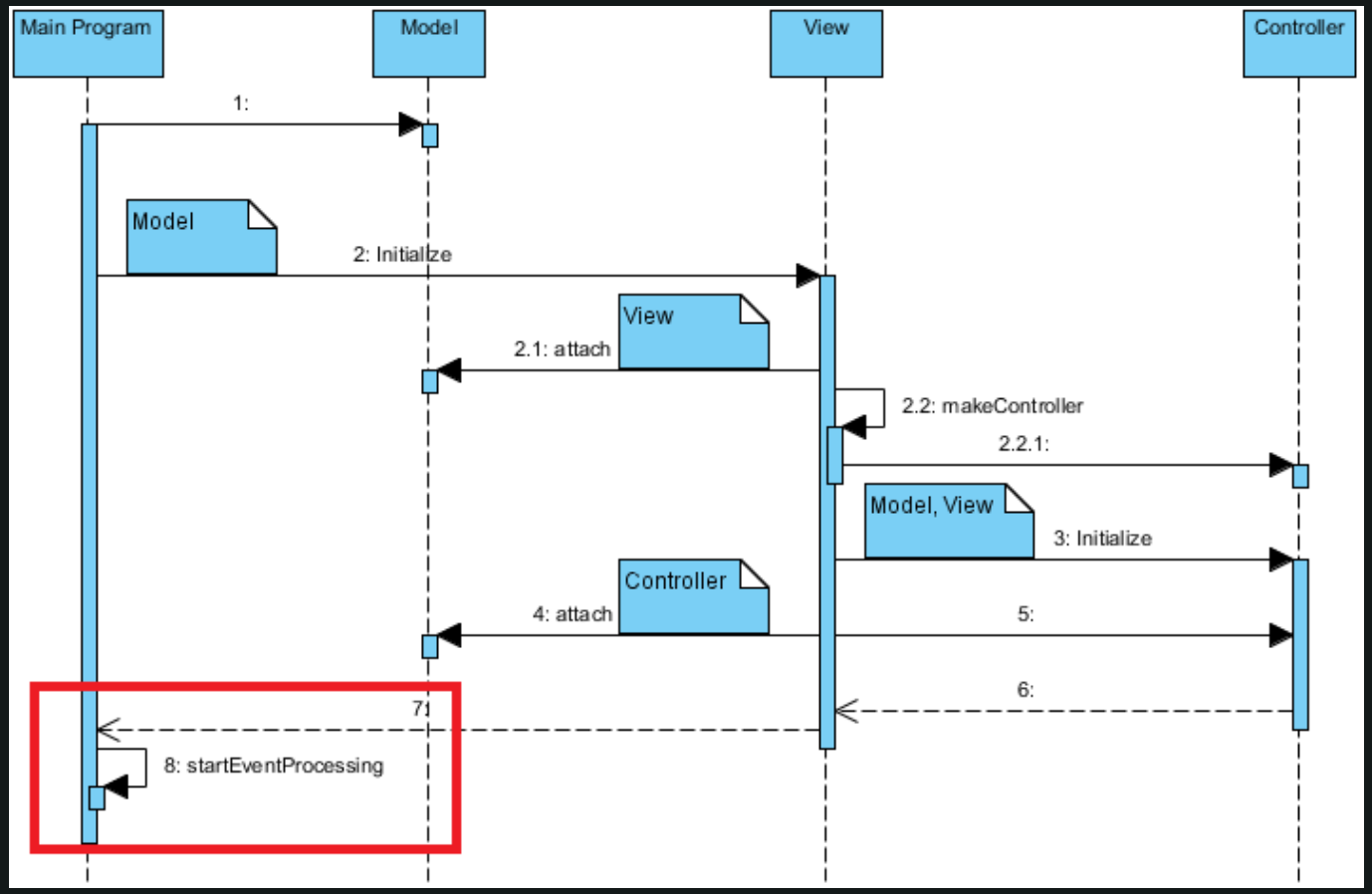
MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 컨트롤러도 attach 프로시저를 호출.
  - 모델의 변경전파 매커니즘에 자신을 등록한다.



MVC Pattern – Dynamic

- MVC 동작 묘사(시나리오2)
  - 초기화 종료.
  - Application은 이벤트를 처리할 준비 완료.



MVC Pattern – Consequence

- 결과(p154~155)
  - 동일한 모델로부터 여러 뷰들을 표시할 수 있다.
  - 모델, 뷰, 컴트롤러의 동기화
  - 뷰와 컨트롤러를 플러그처럼 장착할 수 있다.
  - “룩앤필”의 교환 가능성을 제공한다.
  - 프레임워크로 확장할 수 있다.
  - 복잡성이 증가한다.(단순한 상황에 적용할 경우)
  - 엄청나게 많은 업데이트들이 동시에 진행될 수 있다.
  - 뷰와 컨트롤러가 밀접히 관련된다.
  - 뷰 및 컨트롤러가 모델에 밀접히 결합되어 있다.
  - 뷰가 데이터 액세스하는 일반적인 과정은 비효율적인 경향이 있다.
  - 다른 플렛폼에 이식할때 뷰와 컨트롤러 변경이 필요하다.
  - 최신 사용자 인터페이스 툴에 MVC를 적용하기 어렵다.

감사합니다.