# SMV 소개

Konkuk Univ. IT융합정보보호학과
오예원, 박선영

# 목차

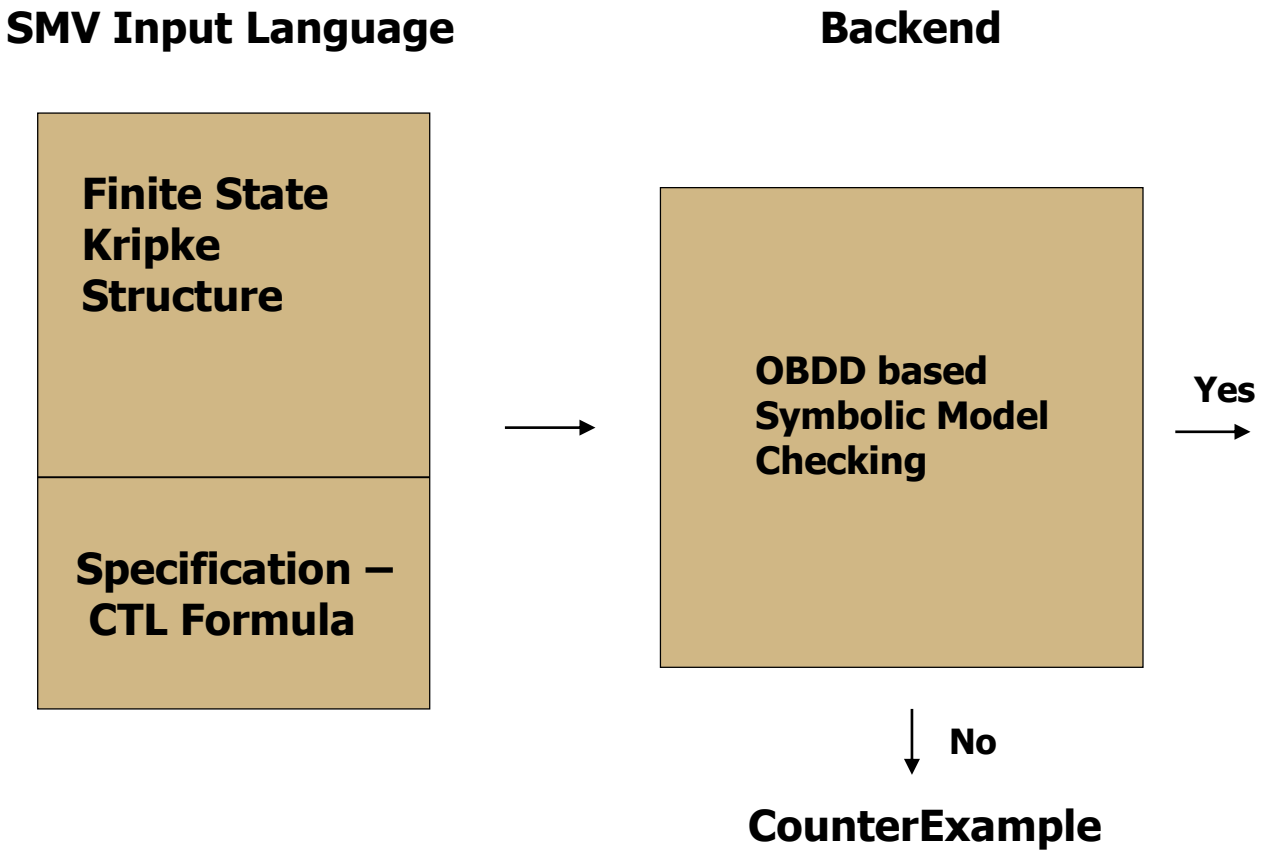
- SMV 소개
- CTL
- NuSMV 설치 방법 및 예시(lift)
- 향후 계획

# SMV

- SMV(Symbolic Model Verifier)는 유한 상태 시스템(finite state system)이 CTL(Computation Tree Logic)이라는 논리와 BDD(Binary Decision Diagram)를 이용하여 요구 명세를 만족하는지를 자동적으로 검증하는 정형 검증 도구.

- CTL을 위한 심볼릭 모델 체킹 알고리즘을 사용.

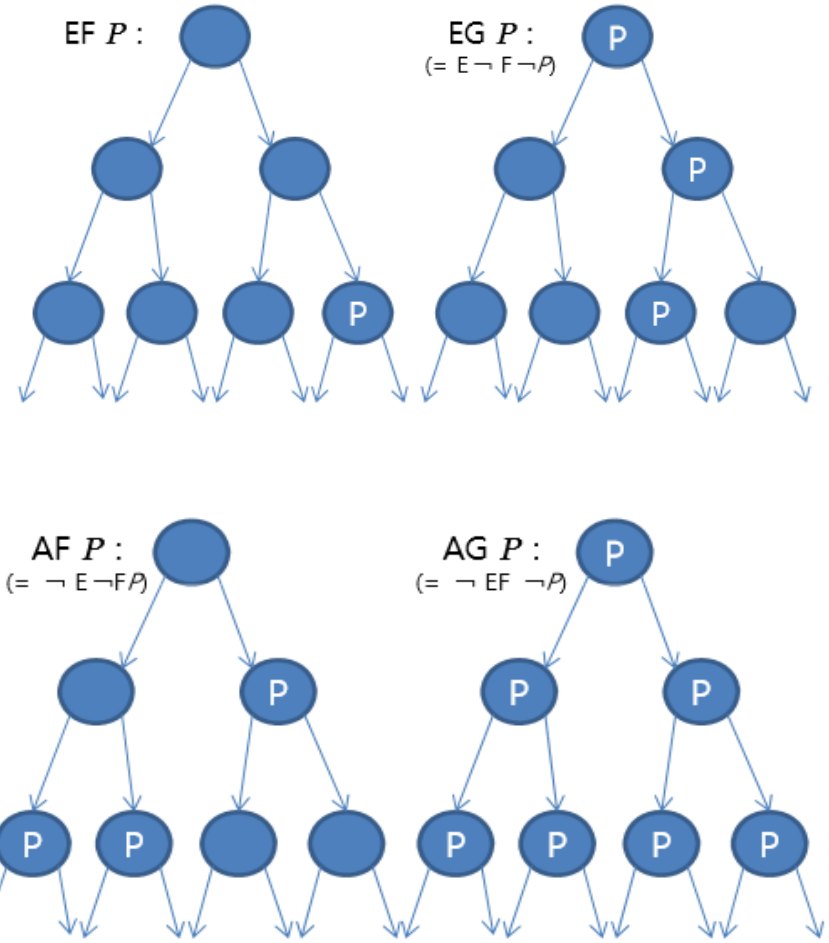
- 모델을 BDD(Binary Decision Diagram)로 표현하고 고정 점 계산으로 속성의 만족성 여부를 판정.



# CTL(Computation Tree Logic)

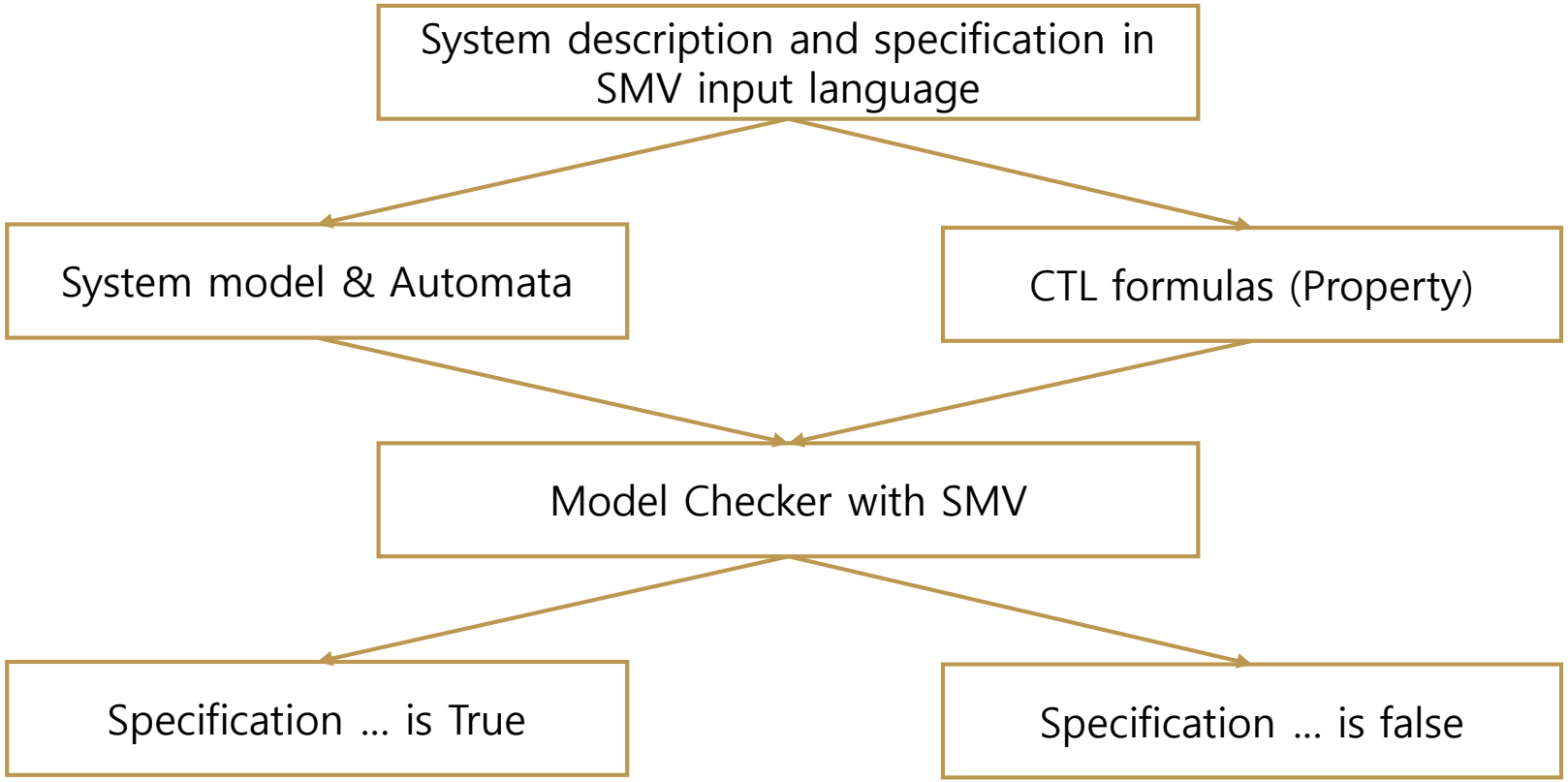
- CTL의 연산자들은 주어진 유한 모델(finite model)의 임의의 한 상태(state)가 주어진 논리식(formula)을 만족하는지 효율적으로 알아 볼 수 있는 fixed point 특성을 가짐.
- 시간의 흐름에 따라 발생 가능한 경로를 트리로 표현
- A - 모든 실행 가능한 경로에 대해

  E - 어떤 실행 가능한 경로에 대해



# SMV System architecture

- Model checking




SMV는 CTL이라는 논리와 BDD를 이용하여 주어진 논리의 참과 거짓을 판별하는 방법이다.

# SMV Variants


- 옛날 버전
- GUI 없음

두개의 버전

- 2.x:
  - 오픈소스
  - 새로운 기능
  - BDD와 SAT 기반
- 1.x:
  - 기존 버전
  - GUI 있음

- GUI 있음
- 새로운 언어 사용

# NuSMV 설치 방법

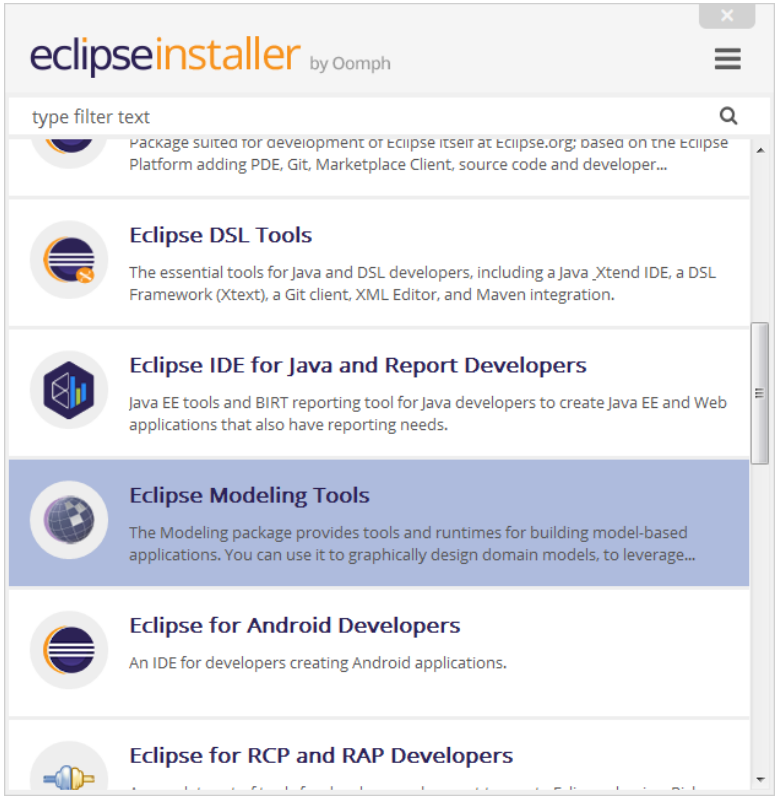
Eclipse에 nuSMV  사용 시 editing을 용이하게 해주는 플러그인 사용.

Eclipse에 Xtext와 nuSMV를 추가 설치.

- **Step1. 프로그램 설치**
  - nusmv 윈도우 인스톨러 설치(2.5.4 windows, 64bit 버전 이용)
  http://nusmv.fbk.eu/bin/bin_download2-v2.cgi
  - Eclipse Modeling Neon 32 비트 설치
  https://www.eclipse.org/downloads/download.php?file=/oomph/epp/neon/R2a/eclipse-inst-win64.exe

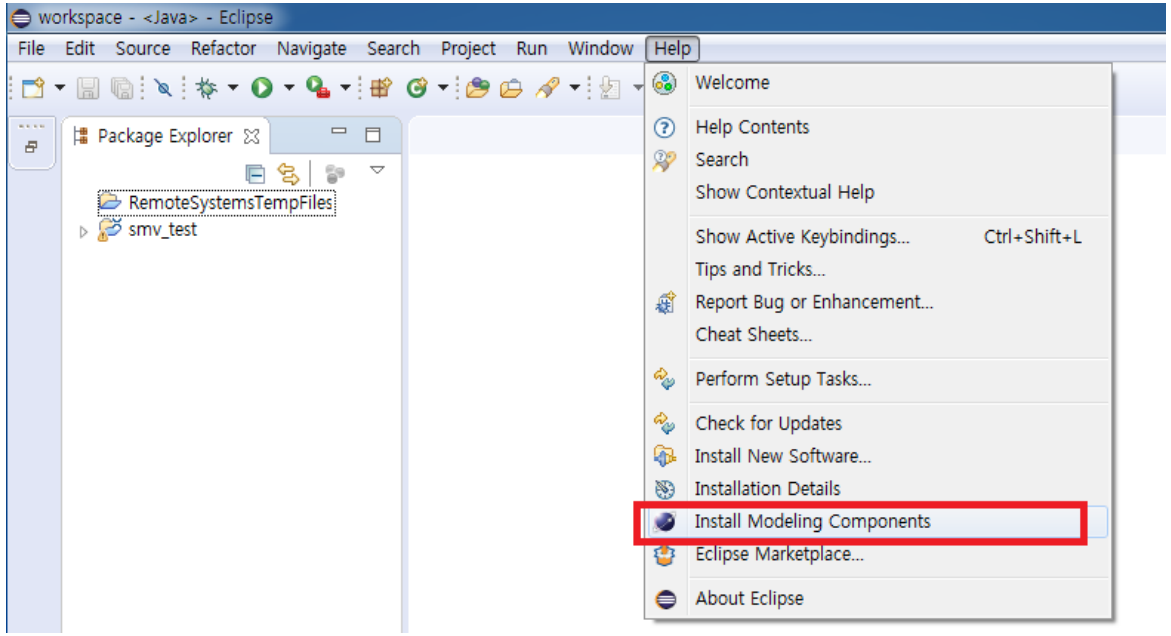
# NuSMV 설치 방법

①




Eclipse Modeling Tools 다운로드

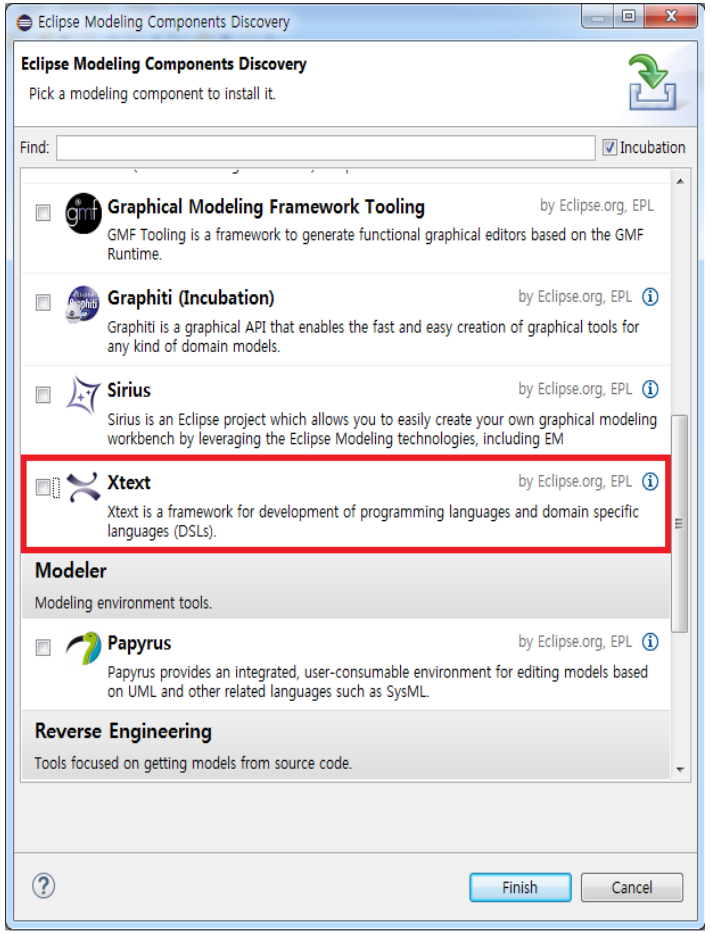
②




Eclipse에서 Xtext를 설치 하기 위해
모델링 컴포넌트 인스톨 수행

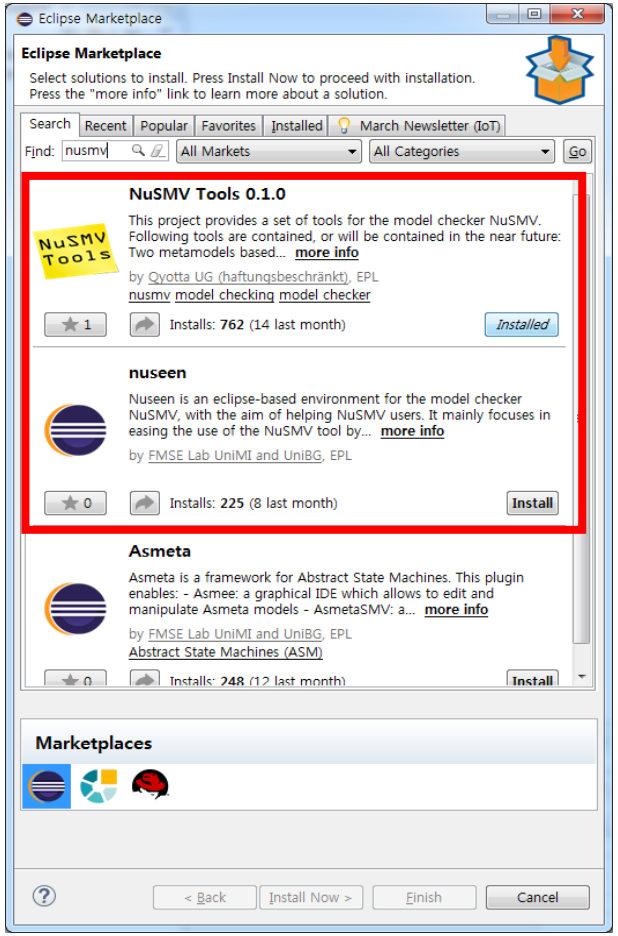
# NuSMV 설치방법

③




Xtext 설치
- edit시 하이라이트 등의 기능을 함.

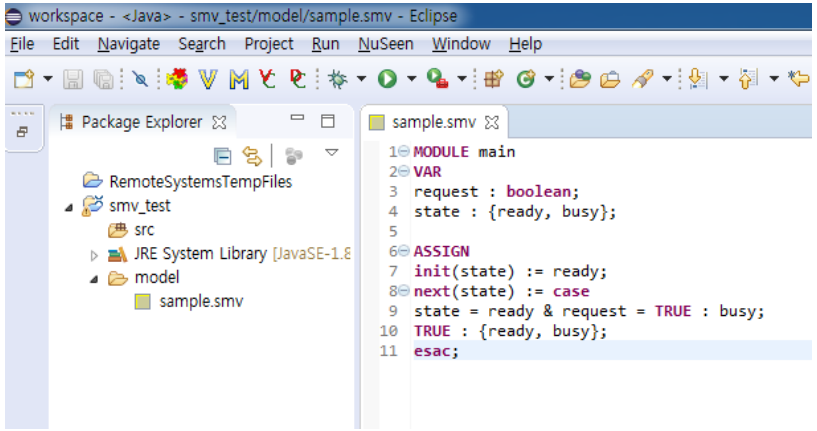
④




NuSMV Tools 0.1.0 & nuseen 설치
- NuSMV 관련 설정을 하기 좋도록 관련 플러그인 설치

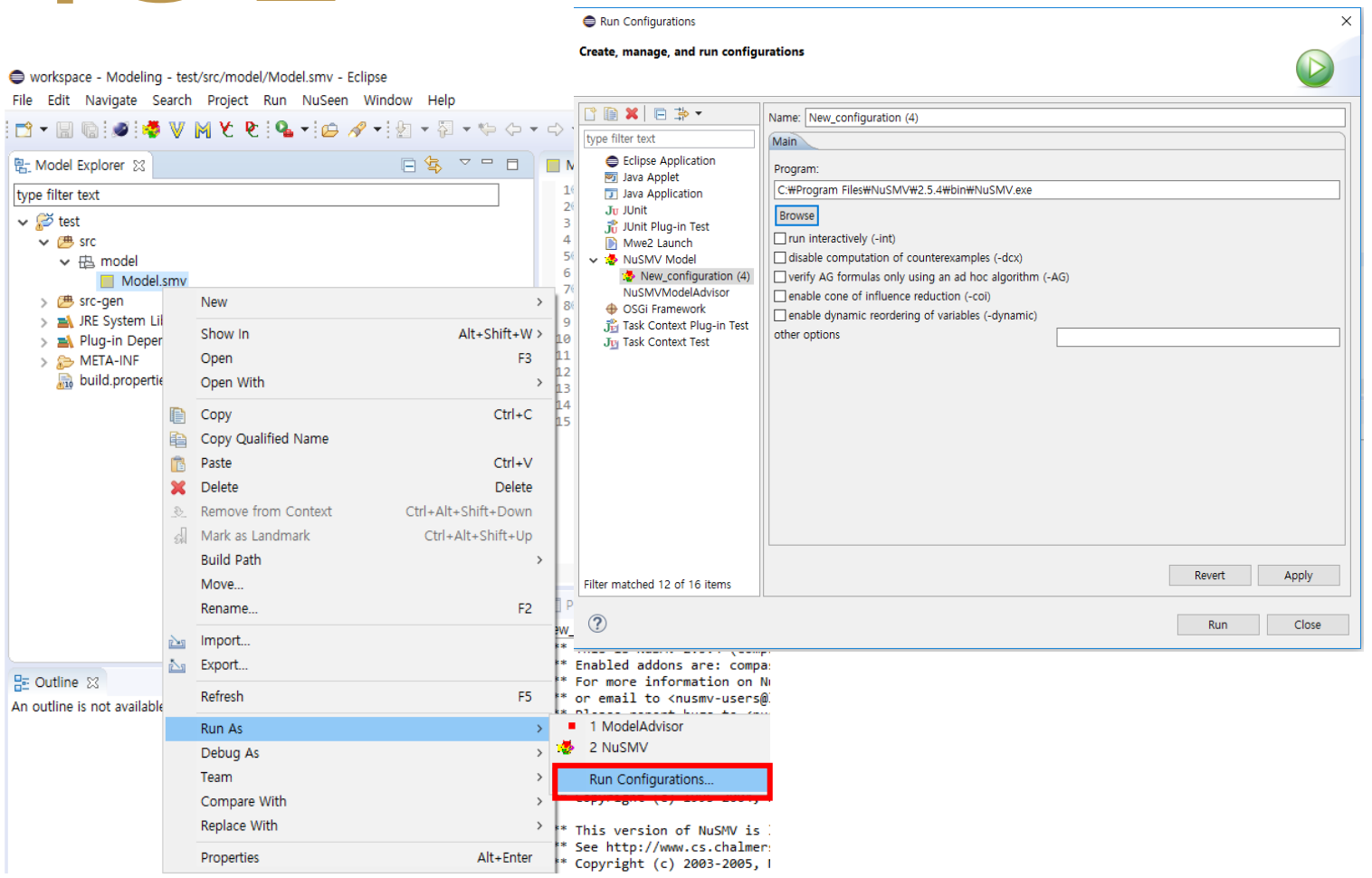
# NuSMV 설치방법

⑤




Xtext project로 프로젝트 생성.

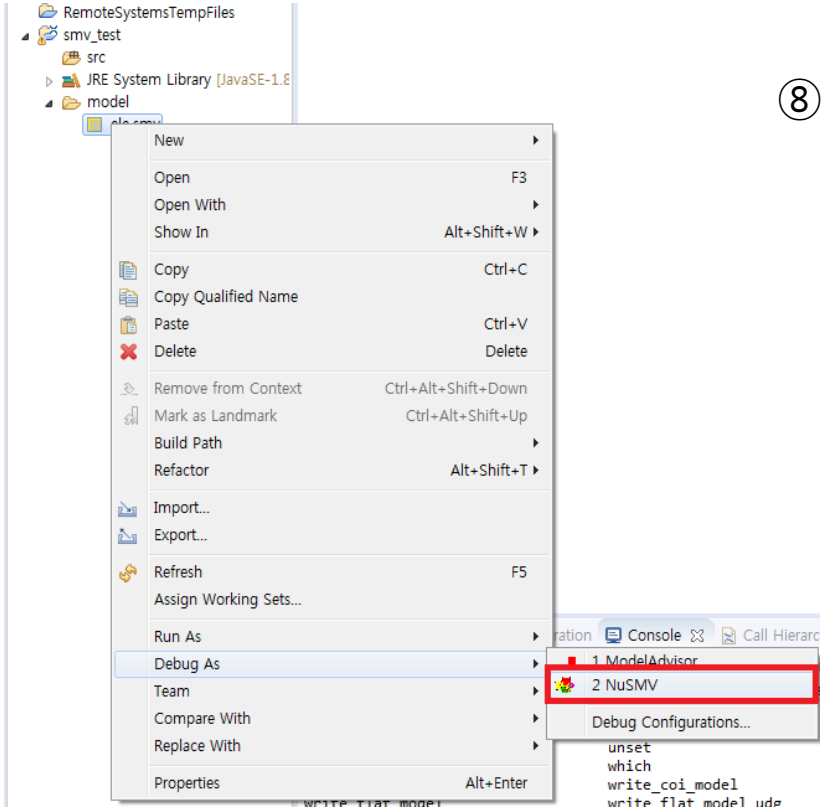
원하는 폴더에 .smv로 파일 생성.




⑥ NuSMV 실행을 위해 Run Configuration 에서 NuSMV Model에 설치한 nuSMV 경로를 지정.

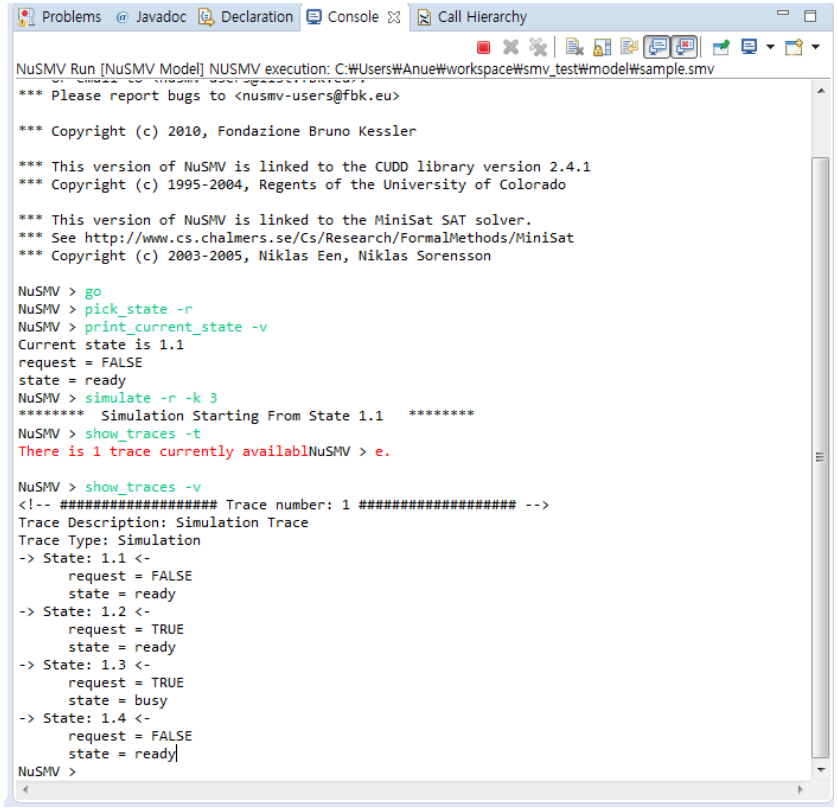
# NuSMV 설치방법

⑦

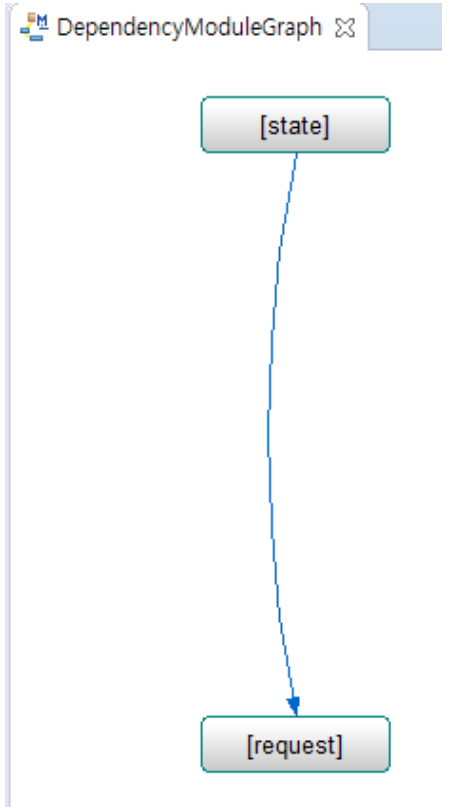



NuSMV를 Run 한다.

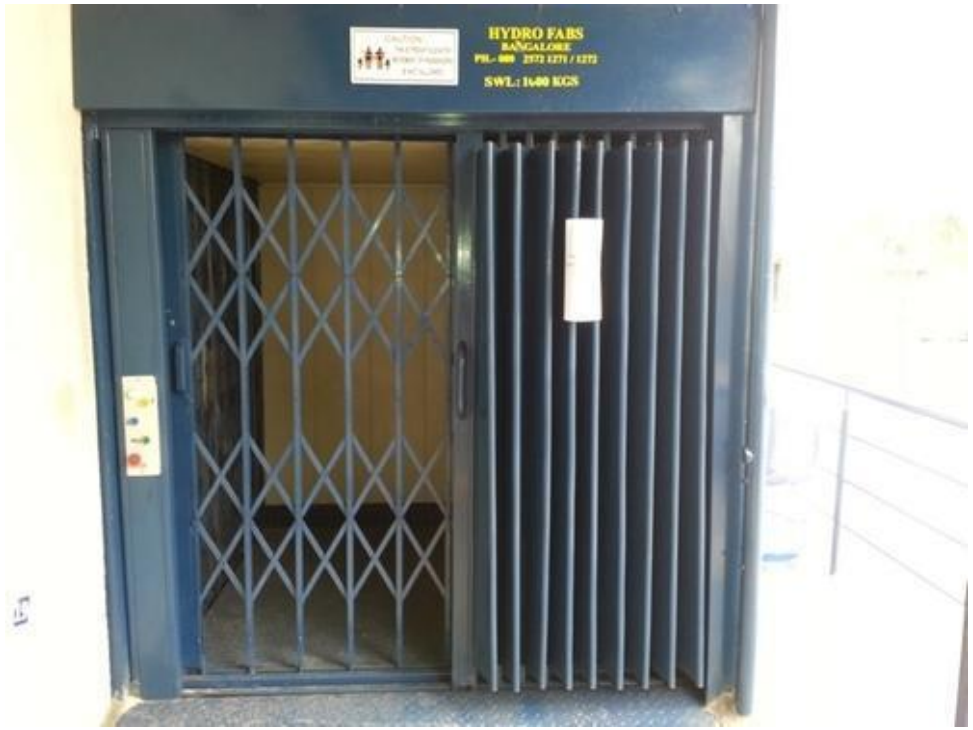
⑧

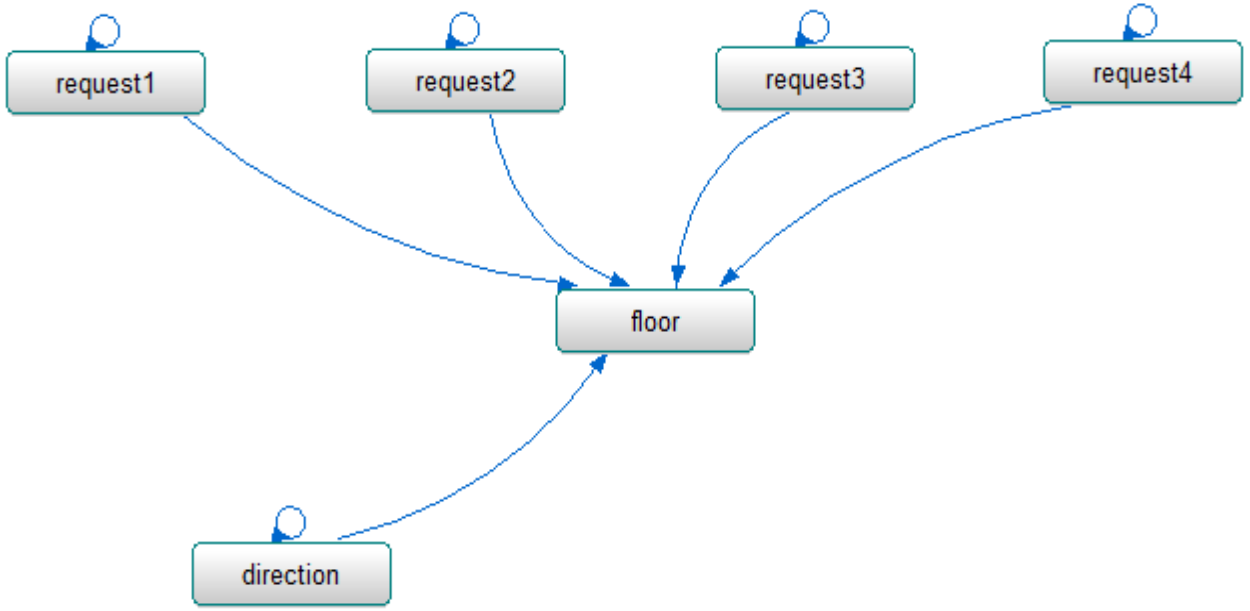



커맨드 창에 명령어 입력을 통해 결과를 확인 가능



# NuSMV 예시 – lift



# NuSMV 예시 – lift

```
MODULE main
VAR
floor : 1..4;
direction: {up , down };
request1 : boolean ;
request2 : boolean ;
request3 : boolean ;
request4 : boolean ;
ASSIGN
init ( floor ) := 1;
init (direction) := up;
init ( request1 ) := FALSE ;
init ( request2 ) := FALSE ;
init ( request3 ) := FALSE ;
init ( request4 ) := FALSE ;
next ( direction ) :=
case
direction=up & floor <4: floor + 1; --올라감
direction= down & floor >1: floor - 1; -- 내려감
TRUE : floor ;
esac ;
next (direction) :=
case
direction=up & next ( floor )=4: down ;
direction= down & next ( floor )=1: up;
TRUE : direction;
esac ;
```

```
next ( request1 ) :=
case
next ( floor )=1: FALSE ; --요청을 삭제하거나(이미1층)
request1 : TRUE ; --요청을 가지고 있거나
TRUE : {FALSE , TRUE };--동작을 수행할 수 있음
esac ;
next ( request2 ) :=
case
next ( floor )=2: FALSE ;
request2 : TRUE ;
TRUE : {FALSE ,TRUE };
esac ;
next ( request3 ) :=
case
next ( floor )=3: FALSE ;
request3 : TRUE ;
TRUE : {FALSE , TRUE };
esac ;
next ( request4 ) :=
case
next ( floor )=4: FALSE ;
request4 : TRUE ;
TRUE : {FALSE , TRUE };
esac ;
SPEC AG !( direction=up & floor =4)
SPEC AG !( direction= down & floor =1)
SPEC AG( request1 -> AF! request1 );
SPEC AG( request2 -> AF! request2 );
SPEC AG( request3 -> AF! request3 );
SPEC AG( request4 -> AF! request4 );
```

# NuSMV 예시 – lift

```
*** This version of NuSMV is linked to the MiniSat SAT solver.
*** See http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat
*** Copyright (c) 2003-2005, Niklas Een, Niklas Sorensson

NuSMV > go
NuSMV > pick_state -r
NuSMV > print_current_state -v
Current state is 1.1
floor = 1
direction = up
request1 = FALSE
request2 = FALSE
request3 = FALSE
request4 = FALSE
NuSMV > simulate -r -k 4
********  Simulation Starting From State 1.1   ********
NuSMV > show_traces -t
There is 1 trace currently available.
NuSMV > show_traces -v
<!-- ################### Trace number: 1 ################### -->
Trace Description: Simulation Trace
Trace Type: Simulation
-> State: 1.1 <-
      floor = 1
      direction = up
      request1 = FALSE
      request2 = FALSE
      request3 = FALSE
      request4 = FALSE
-> State: 1.2 <-
      floor = 2
      direction = up
      request1 = FALSE
      request2 = FALSE
      request3 = TRUE
      request4 = FALSE
-> State: 1.3 <-
      floor = 3
      direction = up
      request1 = TRUE
      request2 = FALSE
      request3 = FALSE
      request4 = FALSE
-> State: 1.4 <-
      floor = 4
      direction = down
      request1 = TRUE
      request2 = TRUE
      request3 = FALSE
      request4 = FALSE
-> State: 1.5 <-
      floor = 3
      direction = down
      request1 = TRUE
      request2 = TRUE
      request3 = FALSE
      request4 = FALSE
```

# 향후 계획

- 모델: Smart Elevator
- 요구사항
  - Door 설치
  - 층수 증가
  - Deadlock 상태 검증
  - Elevator 대수 증가에 따른 우선 순위 구성

# 감사합니다