SPIN

Simple Promela Interpreter

고급 소프트웨어공학

이종원, 이상진

2017.04.03

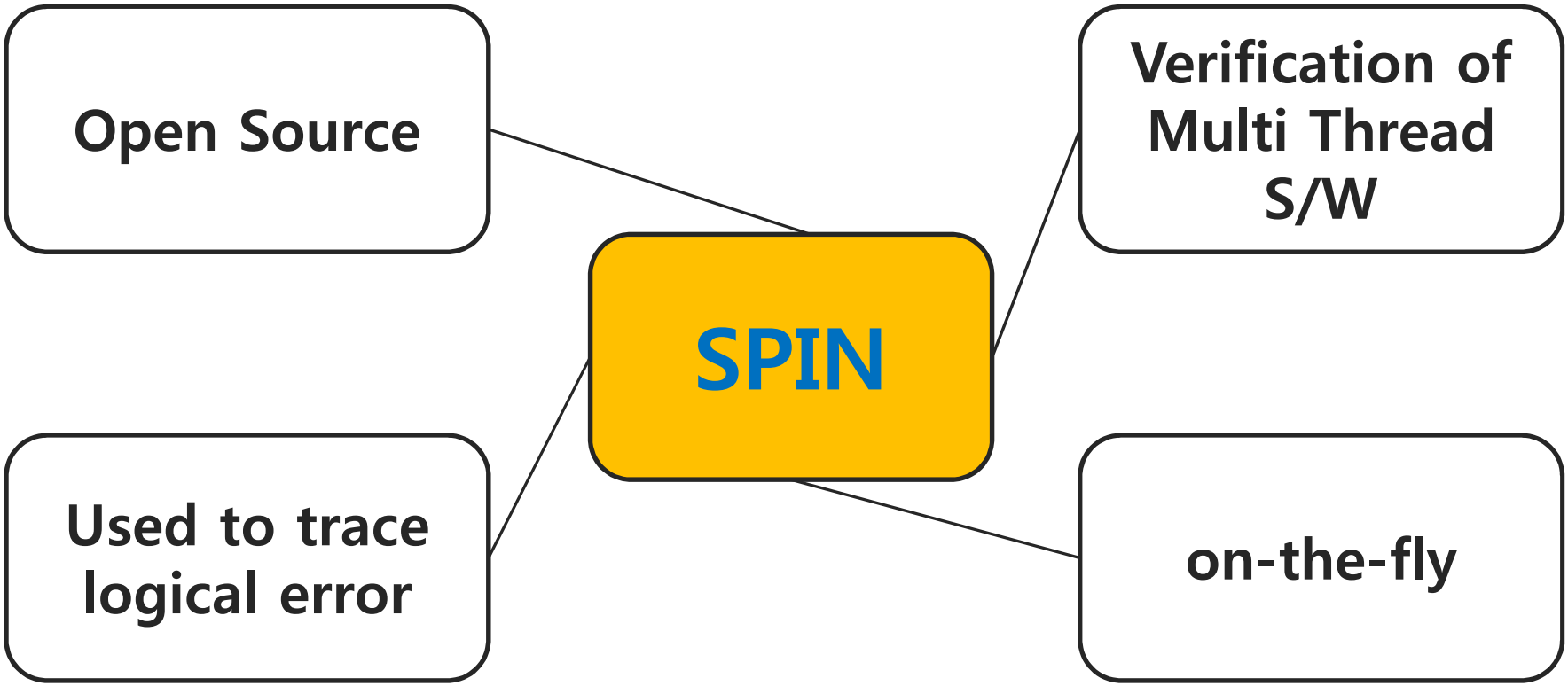
목차

01

Introduce of SPIN

01. Introduce of SPIN

- **SPIN: Simple Promela Interpreter**



✓ 오픈 소스 기반으로, 멀티 쓰레드 소프트웨어의 검증을 위한 도구

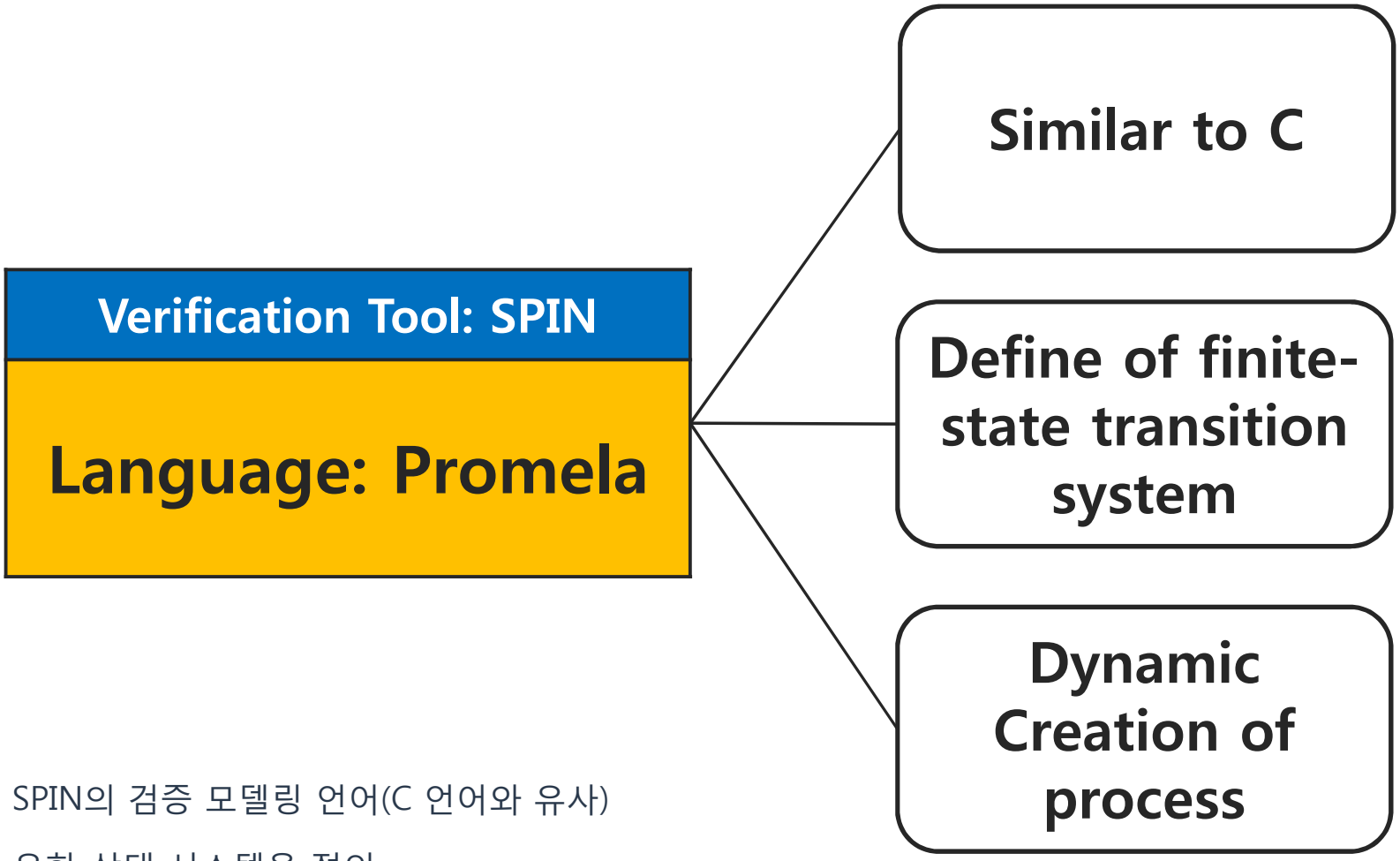
❖ (운영체제, 프로토콜, 분산·제어 소프트웨어 등의 시스템 설계 시) 논리적 오류를 추적

✓ 시스템 확인을 위한 조건을 사전 구성 불필요 ∴즉석으로 동작

✓ LTL 모델 검사 시스템으로도 사용되며, 명시적 상태의 모델을 효율적으로 검증!

01. Introduce of SPIN

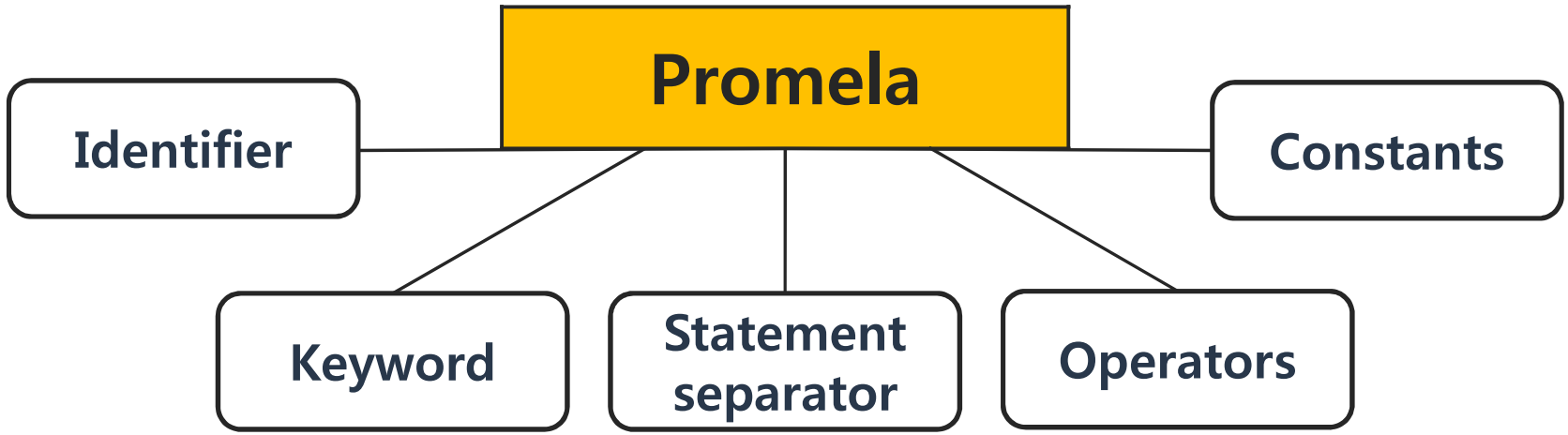
- **Promela:** (**Pro**cess·**Pro**tocol) **Me**ta **La**nguage



- ✓ SPIN의 검증 모델링 언어(C 언어와 유사)
- ✓ 유한 상태 시스템을 정의
- ✓ 프로세스들의 동적 생성 및 메시지 채널을 이용한 메시지의 비 • 동기 통신 모델링 가능

01. Introduce of SPIN

- **Promela:** (**Pro**cess·**Pro**tocol) **Me**ta **La**nguage



✓ 5개의 토큰으로 구성

① 식별자 (Identifier): 단일 문자

② 키워드 (Keyword): bool, goto, int, break, if, fi

③ 상수 (Constants): #define NAME 5

④ 연산자 (Operators): +, -, /, %, !=, ||

⑤ 구문 분리 기호(Statement separator): ->, ;

02

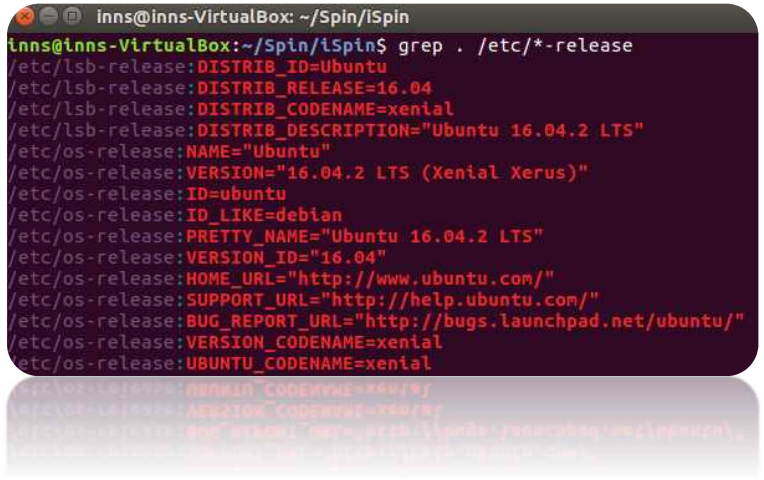
Installation of SPIN

02. Installation of SPIN

- **Process of installation**

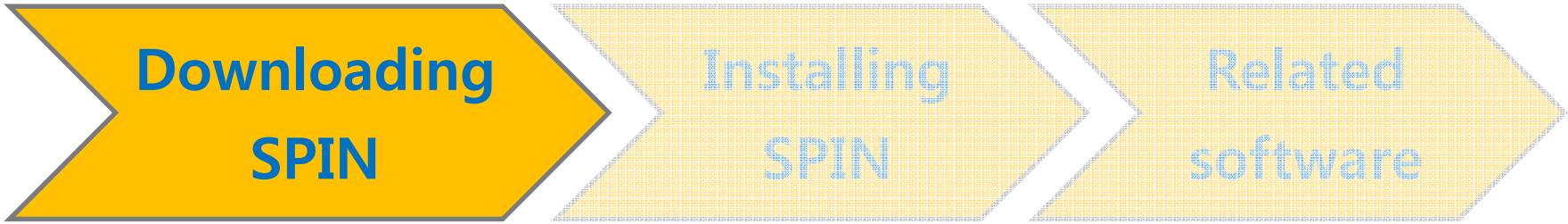
Downloading SPIN → Installing SPIN → Related software

- **Linux**
  - ✓ Oracle VM VirtualBox
  - ✓ Ubuntu(32-bit)



02. Installation of SPIN

- **Process of installation**



1. Downloading SPIN

① $ wget http://spinroot.com/spin/Src/spin646.tar.gz

```
nns@inns-VirtualBox:~$ sudo wget http://spinroot.com/spin/Src/spin646.tar.gz
sudo] password for inns:
-2017-04-02 16:33:15--  http://spinroot.com/spin/Src/spin646.tar.gz
esolving spinroot.com (spinroot.com)... 104.28.11.21, 104.28.10.21, 2400:cb00:2048:1::681c:a15, ...
접속 spinroot.com (spinroot.com)|104.28.11.21|:80... 접속됨.
TTP request sent, awaiting response... 200 OK
ength: 673334 (658K) [application/x-gzip]
aving to: ‘spin646.tar.gz.1’

pin646.tar.gz.1                 100%[=====================================================================>] 657.55K   774KB/s    in 0.8s

017-04-02 16:33:16 (774 KB/s) - ‘spin646.tar.gz.1’ saved [673334/673334]
```

02. Installation of SPIN

- **Process of installation**

2. Installing SPIN

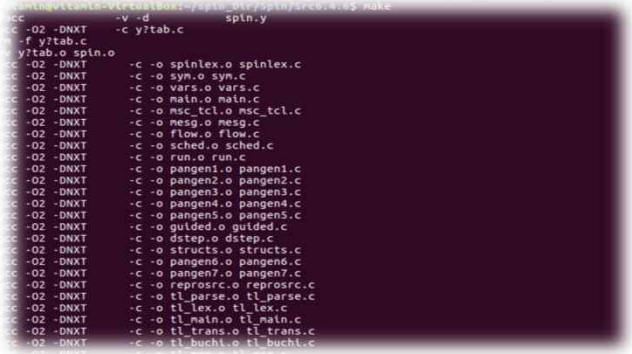
① $ gunzip *.tar.gz

② $ tar –xf *.tar

③ $ cd Spin/Src*

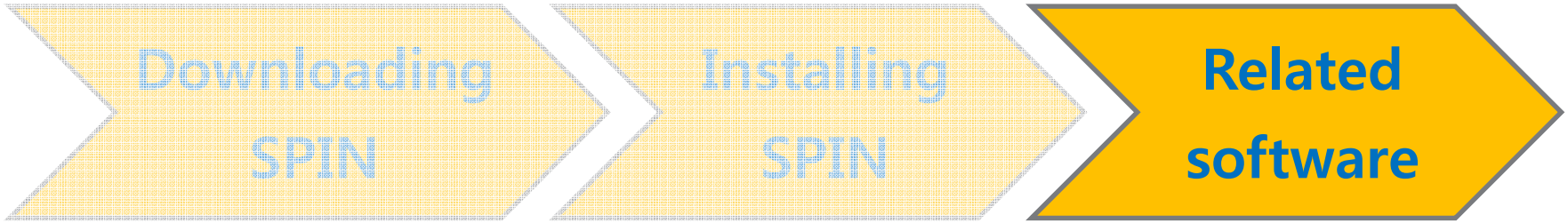
④ $ apt-get install bison

⑤ $ make



02. Installation of SPIN

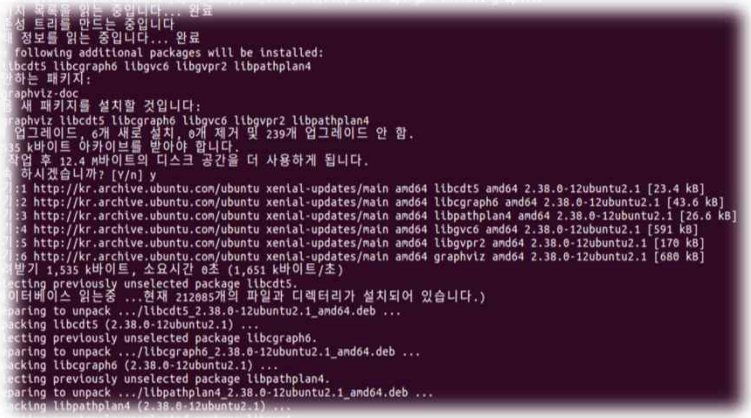
- **Process of installation**

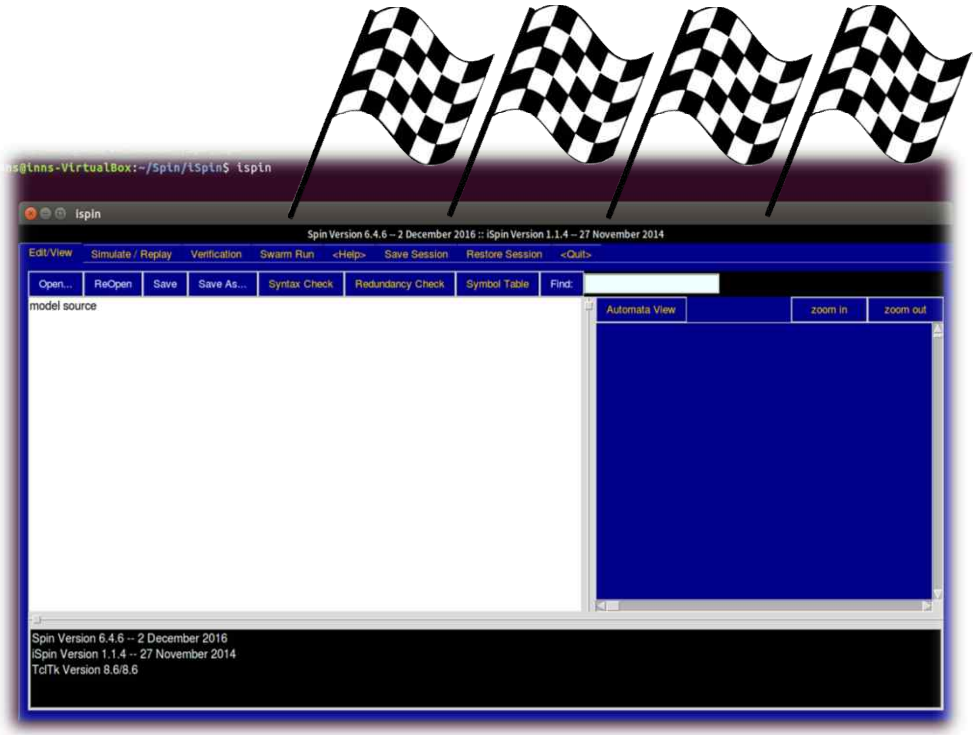


3. Related software

① $ install.sh 실행

② $ apt-get install graphviz



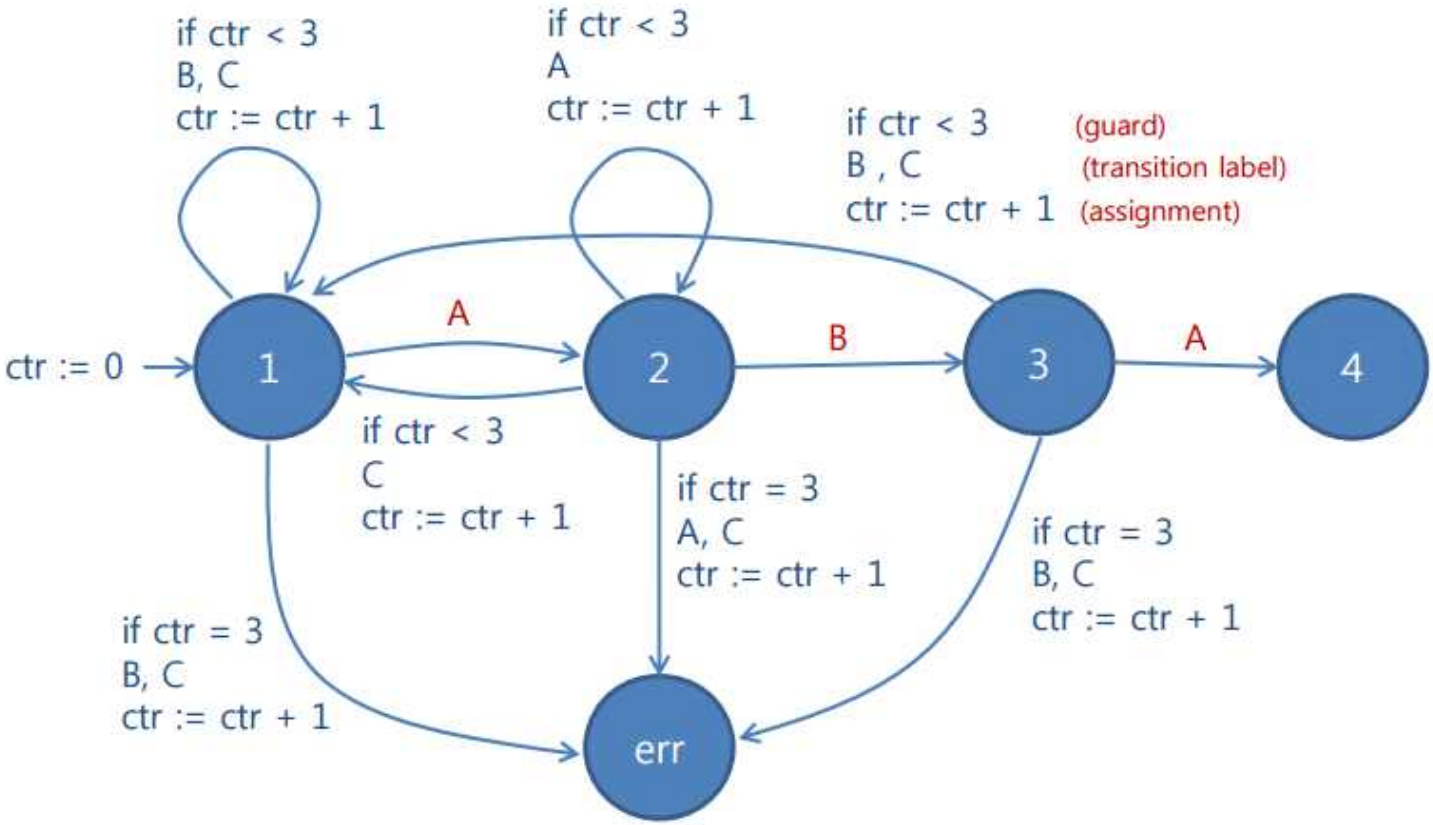


03

Using ISPIN

03. Using ISPIN

- **Verification of digital door lock**



✓ Ref. 2017-1-Advanced Software Engineering-Lecture Note-19p

03. Using ISPIN

▪ Verification of digital door lock

✓ Trial and error

```
#define A 10
#define B 20
#define C 30

int ctr;
int order;
int password[3];
bool state;
mtype = {btn_A,btn_B,btn_C}
chan input_channel = [1] of {mtype}
chan door_channel = [1] of {mtype}

proctype user(){
    if
    ::(state==true)
        if
        ::input_channel!btn_A;
            door_channel?btn_A;
        ::input_channel!btn_B;
            door_channel?btn_B;
        ::input_channel!btn_C;
            door_channel?btn_C;
        fi;
    fi;
}
```

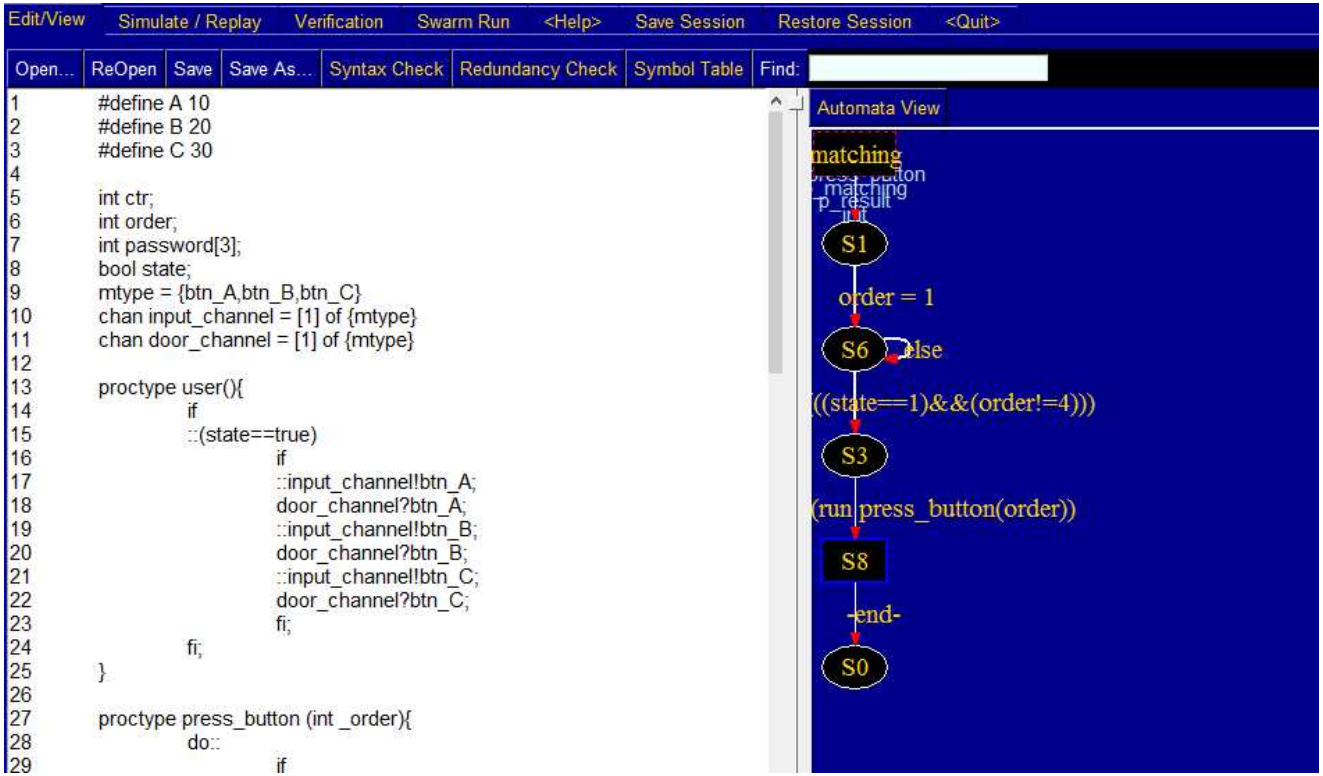
```
proctype press_button (int _order){
    do::
        if
        ::(door_channel!=password[_order-1])->
            order = 1;
            if
            ::(ctr<3)->
                ctr = ctr+1;
            ::(ctr==3)->
                state = false;
            fi;
        ::else->
            order = order+1;
        fi;
    od;
}

proctype matching(){
    order = 1;
    L1: if
        ::(state==true && order != 4)->
            run press_button(order);
        ::else->
            goto L1
        fi;
}
```

```
proctype result(){
    if
    ::(state==false)->
        printf("error");
    ::(order==4)->
        printf("open");
    fi;
}

init {
    ctr = 0;
    state = true;
    password[0] = A;
    password[1] = B;
    password[2] = A;

    run user();
    run matching();
    run result();
}
```

03. Using ISPIN

- **Verification of digital door lock**

✓ Trial and error



03. Using ISPIN

- **Verification of digital door lock**

✓ Code

```
int ctr;
    bool door_open;
    bool press_A;
    bool press_B;
    bool press_C;
    mtype = {btn_A, btn_B, btn_C,unlock,error}

    chan door_state = [1] of {mtype}
    chan input = [1] of {mtype}

    proctype person(){
        if
        ::input!btn_A->
            press_A = true;
            press_B = false;
            press_C = false;
            input?btn_A;
        ::input!btn_B->
            press_A = false;
            press_B = true;
            press_C = false;
            input?btn_B;
        ::input!btn_C->
            press_A = false;
            press_B = false;
            press_C = true;
            input?btn_C;
        fi;
    }
```

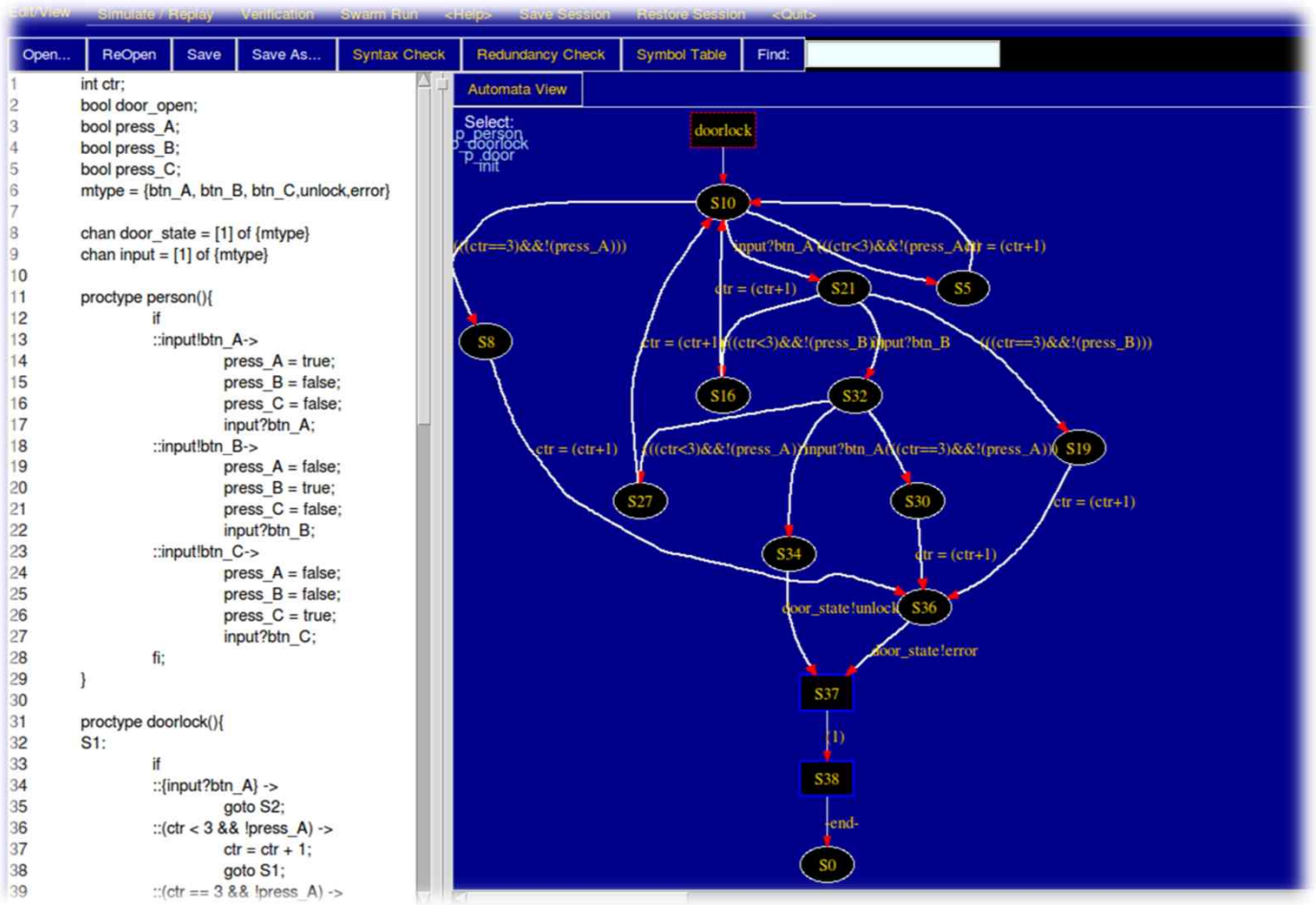
```
proctype doorlock(){
   S1:
        if
        ::{input?btn_A} ->
            goto S2;
        ::(ctr < 3 && !press_A) ->
            ctr = ctr + 1;
            goto S1;
        ::(ctr == 3 && !press_A) ->
            ctr = ctr + 1;
            goto Err;
        fi;

   S2:
        if
        ::{input?btn_B} ->
            goto S3;
        ::(ctr < 3 && !press_B) ->
            ctr = ctr + 1;
            goto S1;
        ::(ctr == 3 && !press_B) ->
            ctr = ctr + 1;
            goto Err;
        fi;
```

```
S3:
        if
        ::{input?btn_A} ->
            goto S4;
            ::(ctr < 3 && !press_A) ->
            ctr = ctr + 1;
            goto S1;
        ::(ctr == 3 && !press_A) ->
            ctr = ctr + 1;
            goto Err;
        fi;

S4:
        door_state!unlock;
        goto end;
    Err:
        door_state!error;
    end:
        skip;
    }

    proctype door(){
        if
        ::{door_state?unlock} -> door_open = true;
        ::{door_state?error} -> door_open = false;
        fi
    }

    init{
        ctr = 0;
        door_open = false;
        run person();
        run doorlock();
        run door();
    }
```

03. Using ISPIN

- **Verification of digital door lock**

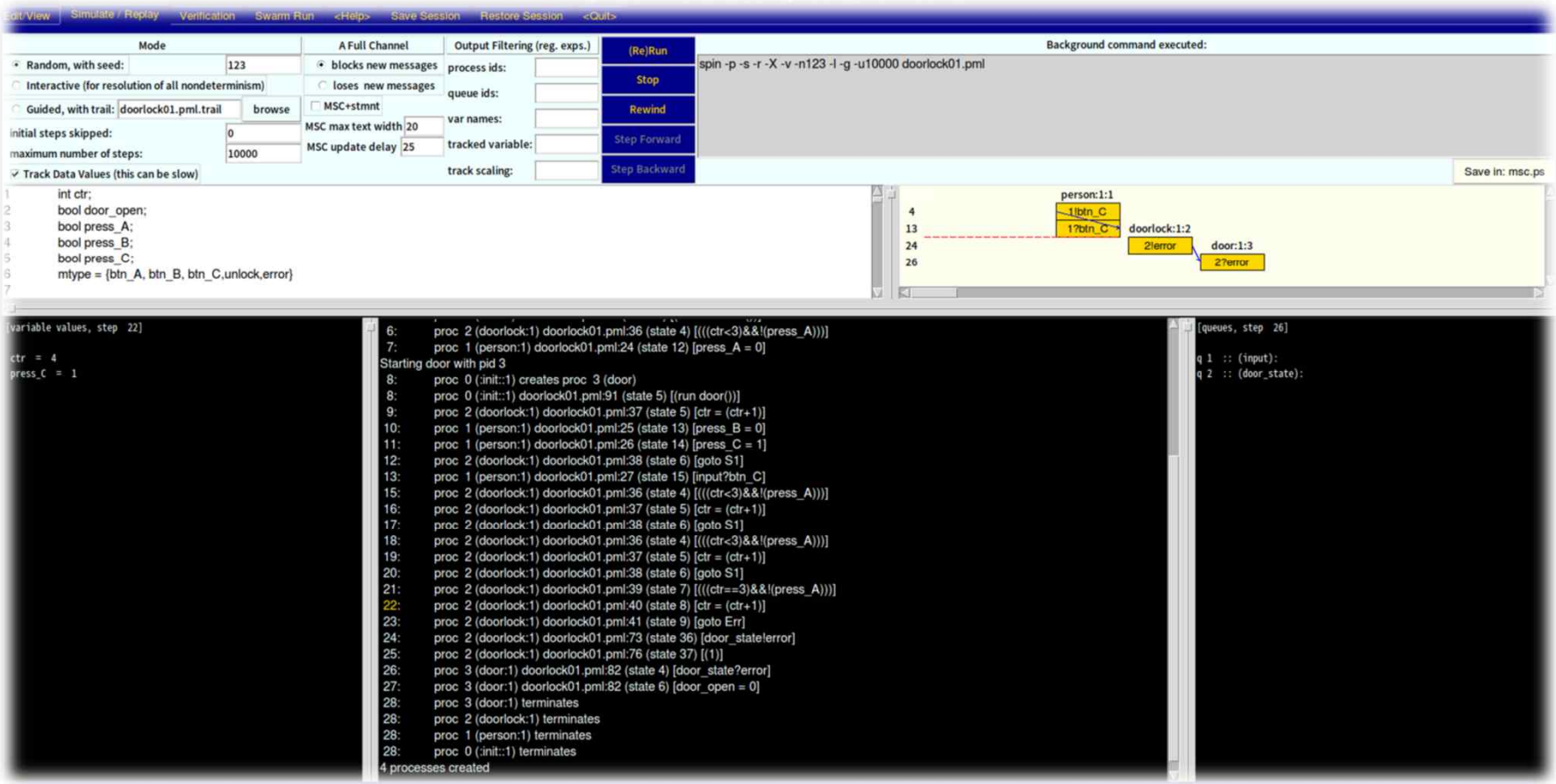
✓ Result - Automata view!



03. Using ISPIN

- **Verification of digital door lock**

✓ Result – Simulate, Run!



?

Q&A