국문 제목

NuSCR 정형 요구 명세에서 UML2.0 Activity Diagram으로의 변환 규칙

영문 제목

Transformation Rules from NuSCR Formal Requirement Specification to UML2.0 Activity Diagram

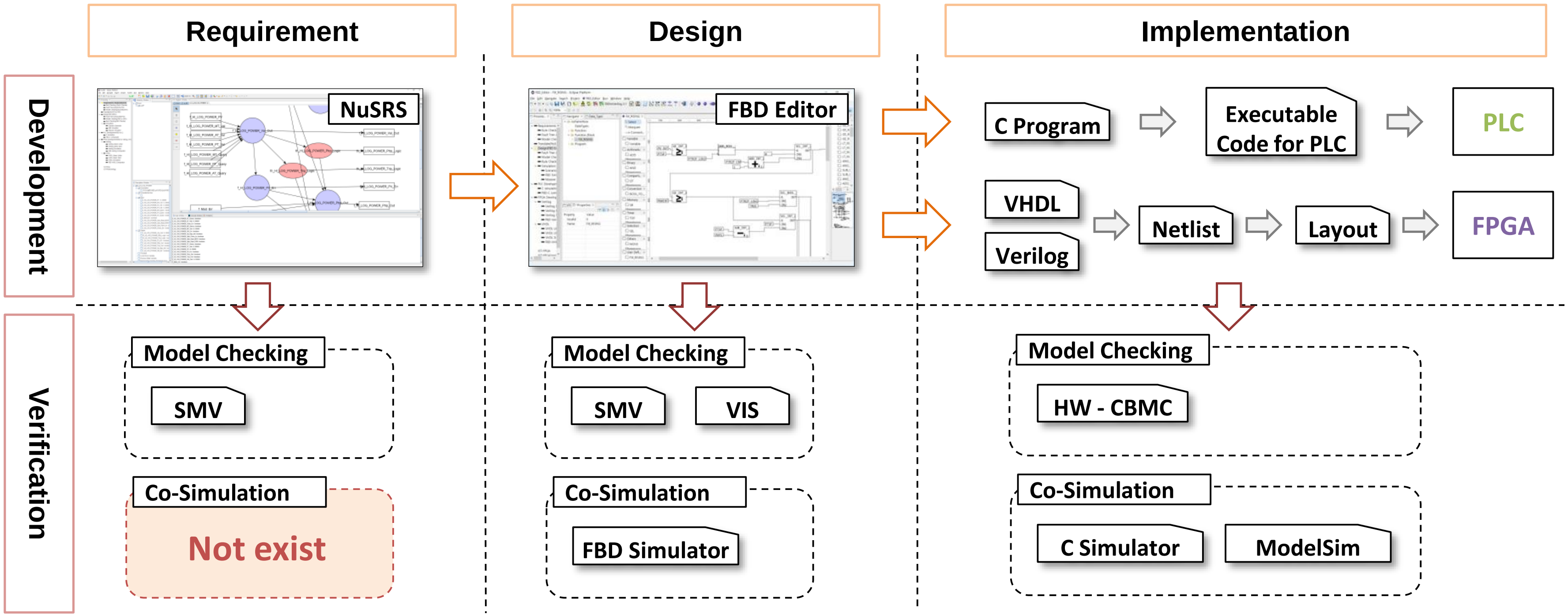
손준익, 정세진, 유준범
Dependable software laboratory
건국대학교 컴퓨터공학과

{sji6227, jsjj0728, jbyoo}@konkuk.ac.kr

Functional Verification of NuSCR

- **Functional verification of NuSCR is important**
  - NuSCR is a formal requirement specification for safety-critical software in NPP (in NuDE 2.0 framework)
  - Detection errors early (requirement phase) ➔ Can reduce costs and increases quality
  - **Model checking** is not enough to check the entire system because of the state explosion problem

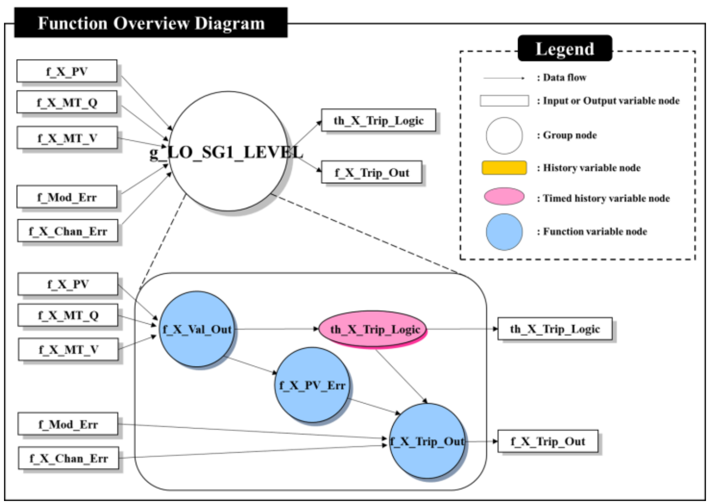
⇨ **We suggest transformation rules from NuSCR to Activity Diagram for the simulation testing**



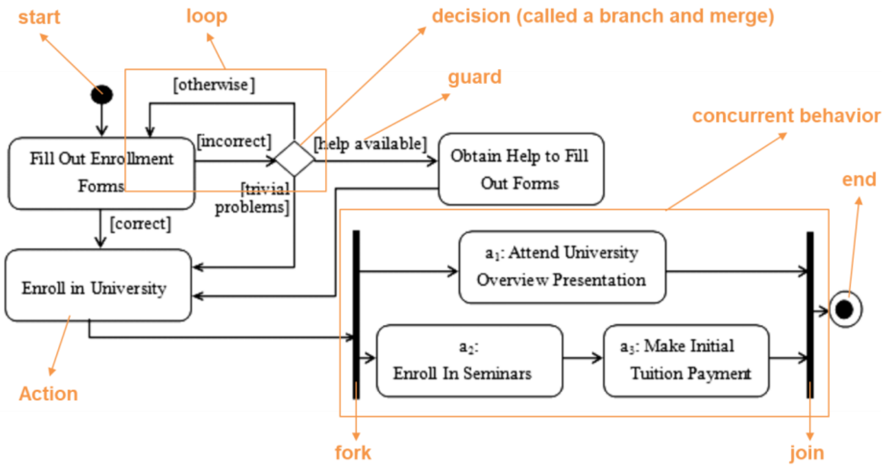
< The NuDE 2.0 framework >

NuSCR & UML 2.0 Activity Diagram

- **NuSCR**
  - Customize SCR to reflect characteristics unique to nuclear engineering domain
  - Parnas four-variable model
    and three basic constructs
    - Function Variable, History Variable, Timed-History Variable (➔ control flow)
  - The relationship of all constructs is represented by FOD (➔ data flow)



- **UML2.0 Activity Diagram**
  - Diagrams depicting the flow of activities step
  - Can be depicting the control and data flow
  - Supporting the decision, loop, and concurrency
  - Used for behavior modeling of various software systems



Transformation Rules
- NuSCR Software System (NSS)

- **NuSCR Software System** : The system specified with NuSCR
  - Using the definitions of all **three basic constructs** and **FOD**
  - Operating periodically with system scan cycle time ***d***.

Transformation Rule 0 (NSS)

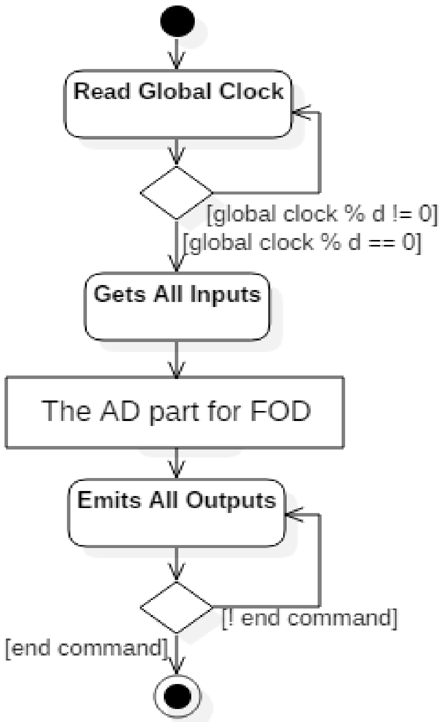
The definition of NSS

$NSS =< S, S_0, R, d >$

- S: a set of system states
- S_0: initial state in S
- R: a set of transition relation $S \times I \rightarrow S' \times O$, where I and O are system's input and output values
- d: system scan cycle time

The behavior of NSS

NSS gets inputs I from the out of system, **calculates with them**, and then emits outputs O to the outside.
➔ Changing its internal system states is FOD



Activity Diagram for NSS

Transformation Rules
- Function Overview Diagram (FOD)

- **Function Overview Diagram** : A kind of DFD, describing the relationship between constructs
  - Composed hierarchically and in this case the group nodes are used
  - All nodes in FOD have partial orders

Transformation Rule 1 (FOD)

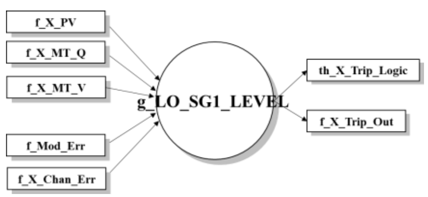
The definition of FOD

$FOD = < N, T >$

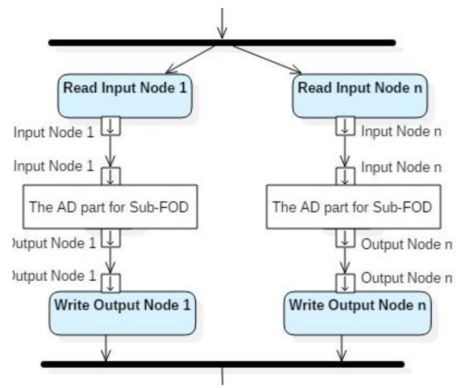
N: a set of all nodes in FOD

T : • a set of transitions (n_1, n_2) between all nodes n_1, n_2 in N

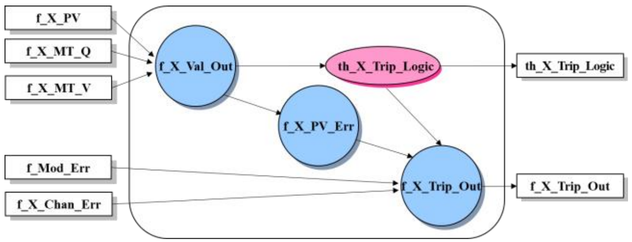
• $\forall t = (n_1, n_2) \in T$, n_1 has precedence on n_2



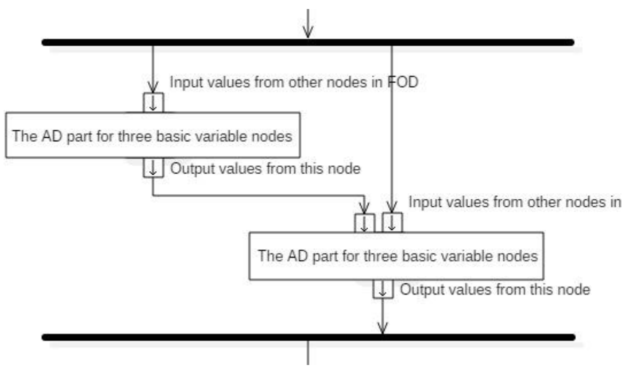
FOD for the hierarchy relationship



Activity Diagram for FOD



FOD for the constructs relationship



Activity Diagram for FOD

Transformation Rules
- Structured Decision Table (SDT)

- **Structured Decision Table** : A kind of Condition / Action table
  - Function variable are used for the mathematical functional behavior of systems and defined as SDT

Transformation Rule 2 (SDT)

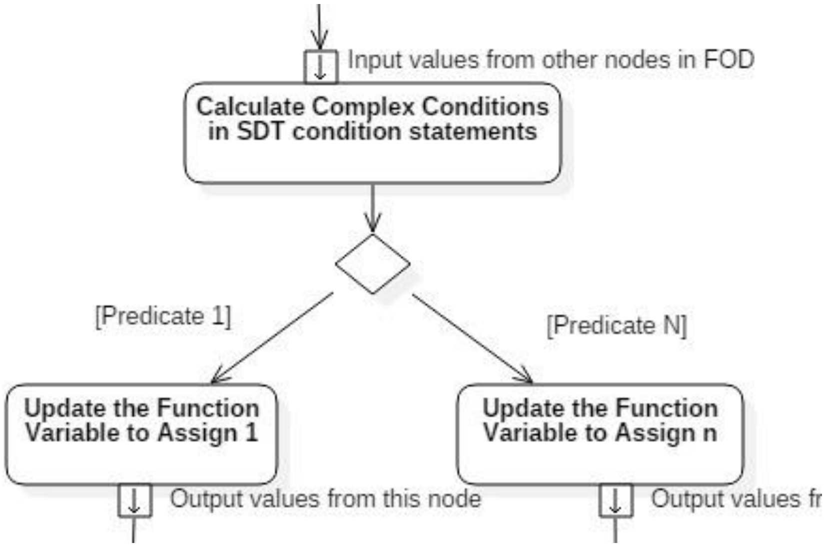
The definition of SDT

SDT: a set of pair (p, a)

- $p \in Predicate$ and a $\in Action$

Conditions		
f_X_MT_Q = true	T	F
Actions		
f_X_V_O := f_X_MT_V	O	
f_X_V_O := f_X_PV		O

SDT for f_X_Val_Out



Activity Diagram for SDT

Transformation Rules
- Finite State Machine (FSM)

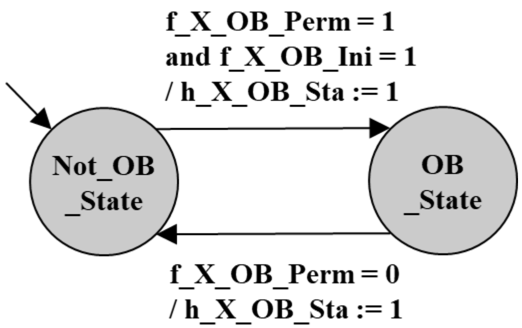
- **Finite State Machine** : Consisting of finite number of states, transitions between states, and labels
  - History variable are used for specifying the state-based behavior of a system and defined as FSM

Transformation Rule 3 (FSM)

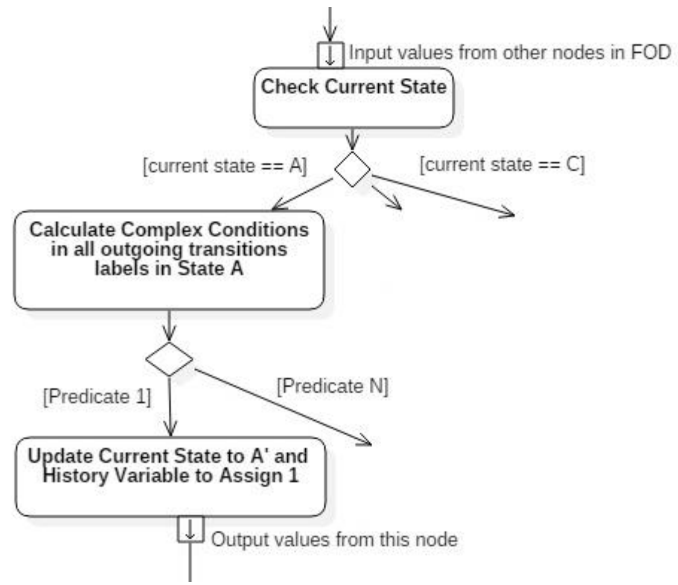
The definition of FSM

$$FSM =< S_H, s_0, C, A, R >$$

- S_H: a set of all states in history variable node
- s_0: initial state in S_H
- C: a set of complex_conditions
- A: a set of assignments
- R: a transition relation $S_H \times C \times A \times S_H$



FSM for h_X_OB_Sta



Activity Diagram for FSM

Transformation Rules
- Timed Transition System (TTS)

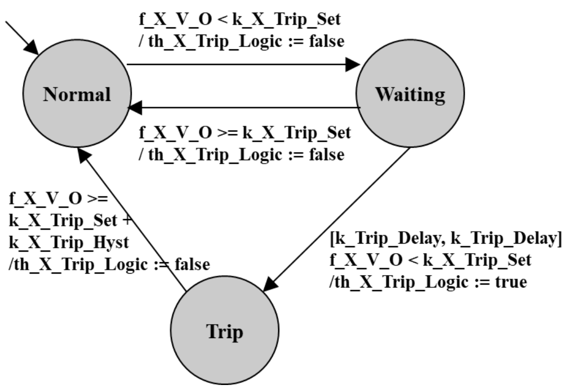
- **Timed Transition System** : An FSM extended with the timing constrains
  - Timed history variable are used for specifying the time-related behavior of a system and defined as TTS

Transformation Rule 4 (TTS)

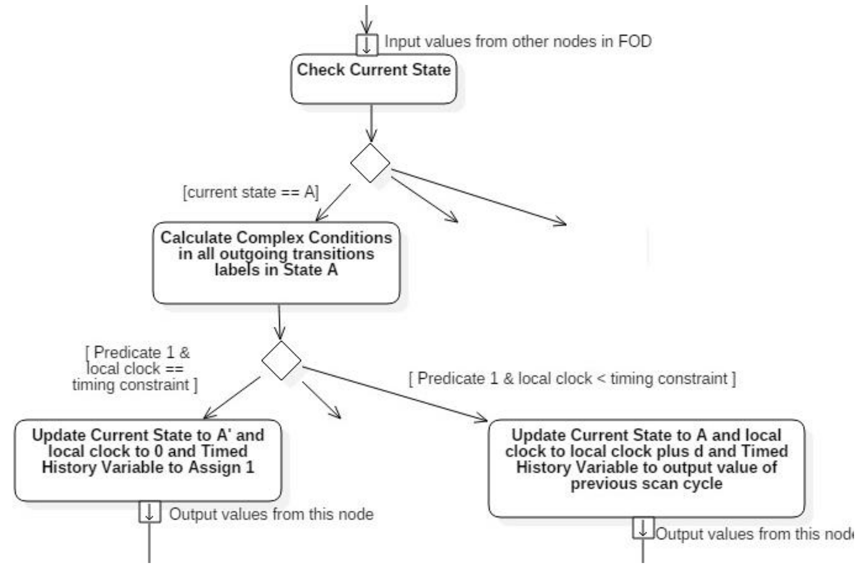
The definition of TTS

$$TTS =< S_{TH}, s_0, C, A, R >$$

- S_H: a set of states in timed history variable node $\times$ lc, where lc is a local clock in LC
- s_0: initial state in S_H
- C: a set of timed_conditions or complex_conditions
- A: a set of assignments
- R: a transition relation $S_{TH} \times C \times A \times S_{TH}$



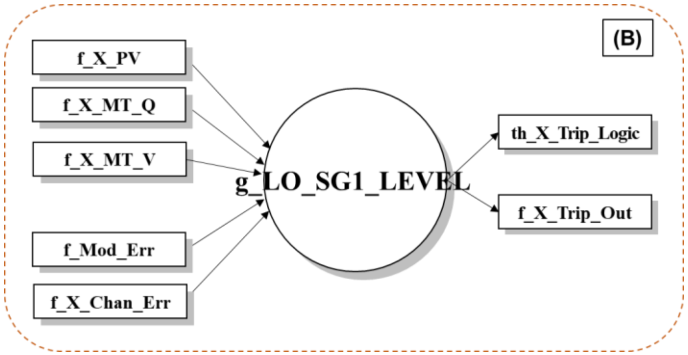
TTS for th_X_Trip_Logic



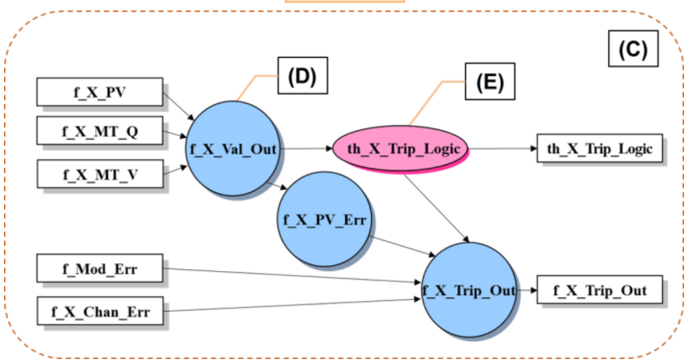
Activity Diagram for TTS

Case Study

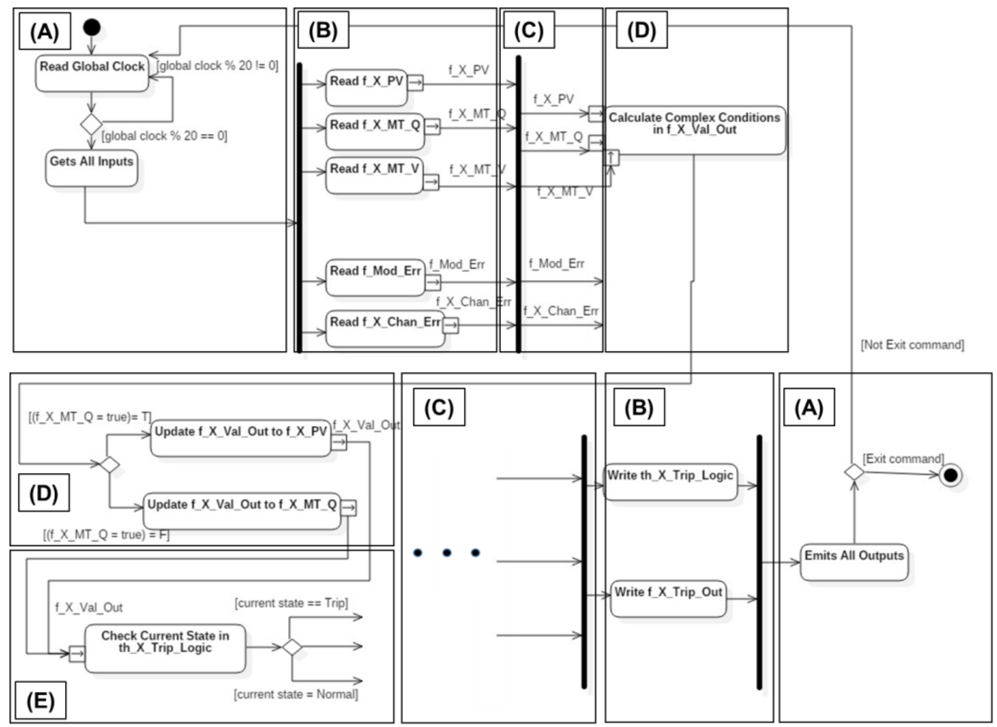
- We performed a case study with some modules of a KNICS APR-1400 RPS BP as an example
  - Target module : g_LO_SG1_LEVEL, which is a fixed falling trip logic



g_BP



g_LO_SG1_LEVEL



Activity Diagram for BP

Conclusion and Future Work

- We suggest transformation rules from NuSCR to Activity Diagram for the simulation testing.
  - The rules were defined using the definitions and behaviors of NuSCR constructs.
- We performed a case study with some modules of a KNICS APR-1400 RPS BP as an example.
- We are planning to
  - prove the correctness of the proposed transformation rules.
  - develop the CASE tool that can mechanically transform from NuSCR specification to Activity Diagram and execute the Activity Diagram for simulation testing.