STATE TOKEN PETRI NET MODELING METHOD FOR FORMAL VERIFICATION OF COMPUTERIZED PROCEDURE INCLUDING OPERATOR’S INTERRUPTIONS OF PROCEDURE EXECUTION FLOW

YUN GOO KIM1* and POONG HYUN SEONG2
1Central Research Institute, Korea Hydro & Nuclear Power Co., LTD
25-1 Jang-dong, Yuseong-gu, Daejeon, 305-343, Korea
2Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology
373-1 Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea
*Corresponding author. E-mail : goodguy@khnp.co.kr

Received March 12, 2012
Accepted for Publication April 13, 2012

1. INTRODUCTION

The Computerized Procedure System (CPS) is one of the primary operating support systems in the digital Main Control Room. The CPS displays procedure on the computer screen in the form of a flow chart, and displays plant operating information along with procedure instructions. It also supports operator decision making by providing a system decision. A procedure flow should be correct and reliable, as an error would lead to operator misjudgment and inadequate control. In this paper we present a modeling for the CPS that enables formal verification based on Petri nets. The proposed State Token Petri Nets (STPN) also support modeling of a procedure flow that has various interruptions by the operator, according to the plant condition. STPN modeling is compared with Coloured Petri net when they are applied to Emergency Operating Computerized Procedure. A converting program for Computerized Procedure (CP) to STPN has been also developed. The formal verification and validation methods of CP with STPN increase the safety of a nuclear power plant and provide digital quality assurance means that are needed when the role and function of the CPS is increasing.

they do not have sufficient V&V guidelines for computerized EOP. Recently, IEEE standard 1786 (2011) addressed guidelines for the design and implementation of CPS [3]. It also indicates that digital system quality assurance and V&V should be emphasized when a CPS is applied to procedures for risk-significant tasks, such as EOPs. Therefore, the V&V of a CP is needed when the operator writes a procedure, and the V&V should include a structural completeness that ensures there are no execution flow errors.

2. CHARACTERISTIC OF COMPUTERIZED PROCEDURE AND OPERATOR INTERRUPTION

The CP in the APR1400 consists of a multi-hierarchical structure entailing Procedure, Step, Instruction, and Evaluation Logic. Execution of a procedure is determined by the main unit of the procedure: a Step and an Instruction. A procedure is executed according to the flow of steps, and each step is completed according to the instructions within. There are two types of instructions: unitary instruction that has one way of flow, and binary instruction that has two ways of flow. Also, the evaluation logic for instructions consists of both a manual evaluation, where the operator inputs the evaluation result, and an auto evaluation, where the system determines the evaluation result. Each evaluation result and type of instruction affects the execution flow of an instruction. Y. Jung et al. (2004) have developed a model for plant operating procedure with a flowchart and a success logic tree [4]. However, the modeling only deals with execution and display of the CPS and they do not cover verification of the execution of procedures. A simplified normal procedure execution flow is shown in Fig. 1. When the procedure is started, the first step and first instruction of that step start. In a normal flow, a step is completed when the completion of an instruction in the step is connected to the next step. When a step is completed, the next step is started. When the step that is linked to the end of procedure is completed, the procedure is completed. However, there are alternative flows such as “postpone”, “re-execute” and “change flow”, as depicted in Fig. 2. An operator can start a next step before a current step is completed. In other words, an operator can postpone the completion of a current step and start the next step. The postponed step is still being executed. This means that the operator executes the postponed step and the next step simultaneously. Moreover, a completed step can be re-executed when requested by the operator, and the procedure flow can be changed according to the result of the re-execution. Also, an operator can start any step that is not started by “change flow”. As explained, a procedure has a predefined execution flow, and plant information that is used to determine that flow. Therefore, the execution flow of a procedure can be changed by the operator or system, and verification and validation of the execution flow are needed when the procedure is being written.

3. MODELING OF COMPUTERIZED PROCEDURE WITH COLOURED PETRI NET

3.1 Introduction to Coloured Petri Net

A Coloured Petri net is one of the formal and graphical languages for modeling a system. Many computer tools for the Coloured Petri net have been proposed (Kurt Jensen 1997, Kurt Jensen et al 2007) [5,6]. The Coloured Petri net combines the strengths of an ordinary Petri net with the strengths of a high-level programming language. A Coloured Petri net also has a formal and mathematical representation with a well-defined syntax and semantics. It is possible to formulate a standard request for verification.
3.2 Coloured Petri net modeling rules for Computerized Procedure

In modeling CP with a Coloured Petri net, steps and instructions are modeled as places, and the processes of steps and instructions are modeled as transitions. The processes of a procedure are controlled by tokens. Coloured Petri net modeling rules for CP are presented as follows. 1) Every step has one additional place for a starting state. 2) Every step has one additional place for the completion of a step and has a transition to the starting place of the next step. When a step has more than one completion path, each completion path has one place. When a step has a completion path to the end of the procedure or to another procedure, there is no transition, and the procedure execution flow is ended. 3) Unitary instruction is modeled to a place that has one transition. 4) Binary instruction is modeled to a place that has two transitions. 5) When an instruction has sub instructions combined with “OR” logic, each sub instruction is modeled as a place that has its own transition. 6) When an instruction has sub instructions combined with “AND” logic, each sub instruction is modeled as a place, but all sub instructions have a common transition. 7) When an instruction has sub instructions, a “join place” is added that joins the results of the sub instruction. 8) An instruction that has contingency flow is modeled to a place that has two transitions. One transition is connected to the place of next instruction and the other transition is connected to the place of instruction in contingency flow. 9) The last instruction in the contingency flow is modeled to a place that has a transition to the original instruction for this contingency flow. 10) When an instruction that has a contingency flow has sub instructions, the last instruction in the contingency flow is modeled to a place that has a transition to the “join place” of the original instruction that has this contingency flow.

The Coloured Petri net modeling result for a simple procedure is depicted in Fig 3. When the step is started, the token is transited to instruction 1. When the evaluation of the instruction is “YES” then the token is transited to instruction 2. When the evaluation of instruction 1 is “NO” then the token is transited to instruction 3. Instruction 2 and instruction 3 have a transition to the place of step completion that also has a transition to the next step.

3.3 Limitation of Coloured Petri Net Modeling for Computerized Procedure.

Coloured Petri net can model the normal process of the procedure as explained in Section 3.2. However, when the operator postpones the completion of the current step at instruction 1 of step 1, then step 2 should be started. This means that the token from instruction 1 of step 1 should be transited to step 2, and one additional token
should be left in the instruction of step 1. When a postponed step is completed, the token should not be transited to the next step, because the next step has already started. In other words, the transition between the completion of a step and the starting of a step shall not be fired when the next step has already started. Therefore the token will be left in the current step. Moreover, the operator can postpone at any instruction. So, every instruction needs a transition to the next step for the postponed execution flow.

In the Coloured Petri net modeling in Fig 4, there are 3 additional transitions for “postpone”. In the same manner, “re-execution” and “change flow” can be modeled by Coloured Petri net, as also depicted in Fig. 4. When an operator changes procedure flow, then the token at the current instruction should be transited to the first instruction of the changed step. Every instruction needs transitions to all steps in the forward direction. When an operator re-executes an already completed step, then the token at the current instruction step should be transited to the first instruction of the re-executed step. Every instruction needs transitions to all steps in the backward direction.

4. MODELING OF COMPUTERIZED PROCEDURE WITH STATE TOKEN PETRI NET

4.1 Definition of State Token Petri Net

A State Token Petri Net (STPN), which is an extended Coloured Petri net, has been proposed in this work. STPN uses a state machine for the dynamic color of a token. In STPN, a token has states that can be modeled by the state machine. This means that a token has dynamic instead of static color. The state of the token makes it possible for a place to have many states without any transition. Hence, the dynamics of the system can be modeled in two independent ways: transition of place in the Petri net model, and the state machine of the token. The characteristics of STPN are as follows. 1) A token has states in the form of a color set. 2) Token states act like a state machine. 3) An event in a state machine can change another token state. 4) The state transition of a token is independent of the transition of place.

In STPN, a token has a state machine and the current color of the token is the current state of the state machine. STPN modeling, and the example of relations between token and transition are depicted in Fig. 5. Transition B has a guard condition, and it only fires when the state of token B is state 4. The state of token B is changed when there is an event B, which is generated when the state of token A is changed from state 1 to state 2. This means that when there is an event A in the first Petri net, the transition in the second Petri net is fired. Therefore, STPN can model both the normal process and the alternative flow caused by the event. STPN simplifies the normal process part of the model and makes it easier to understand. Also, the state machine of the token in STPN is a simpler modeling language as compared to a Coloured Petri net. Hence, the simple part of the total model can be modeled as a state token, thereby making total modeling simpler than with an ordinary Coloured Petri net. STPN is expected to be useful in the following cases of modeling.

4.1.1 System with Interruptions

A token in STPN carries dynamics that can apply to the every place in the system. In other words, if a place has a token, then the place also has a dynamic set of states that is modeled in the token. One familiar example of this dynamics is an interruption. The user can interrupt the execution independently with the state of the system, and when the interruption is over, the system should return to its original state before the interruption. The state of the token can be modeled as an interruption. Then, every place
that has a token can be interrupted and, moreover, the interruption that can be transferred along with the Petri net depending on the type of interruption.

4.1.2 System with Multi-dynamics

More than one token can carry dynamics. A Petri net was originally used for concurrent system modeling; however, STPN can be used for multi dynamic systems that have concurrent execution flows in the Petri net, and another independent dynamic system as state machines in a token.

4.2 Modeling of Computerized Procedure with State Token Petri Net

The CPS has been modeled with STPN. Steps, instructions, and the processes of a procedure are modeled as ordinary Petri nets. However, the state of a step is modeled as the state of the token, as described in Section 4.1. Every step in the computerized procedure has one of 3 states: "Before Execution" "Under Execution" and "Completed". Initially, the first place in all steps has a token with a state of "Before Execution". When a procedure is started, the state of the token of the first step is changed to "Under Execution" and the transition can be fired. When the operator finishes all instructions, the token arrives at the place named Complete. When the operator completes the step, the state of the token changes from "Under Execution" to "Completed" and the event is generated that changes the state of the token in the next step from "Before Execution" to "Under Execution". This means that every completed step has a token with a state of “Completed”. The completed step can be re-executed when requested by the operator. The state of the token changes from "Completed" to "Under Execution" and it transits to the first place of the step. During "Under execution" the operator can postpone the completion of the current step. The event that changes the state of the token in the next step from "Before Execution" to "Under Execution" is generated.

Emergency Operating Procedures (EOP) are used when there is an accident at a nuclear power plant. The purpose of EOP is to mitigate the accident, and to make the plant safe and stable. The EOP should be performed quickly and accurately, and thus there should be no error in the sequence of procedures in the EOP. Table 1 shows the three major types of steps in an EOP. There are 2 columns, one for expected flow and the other for the contingency flow that is performed when the expected flow cannot be applied. Step 1 has an entry condition, and instructions that are executed when the condition is met. When the pressurizer pressure is less than the set point, the operator must verify that SIAS (Safety Injection Actuation System) is actuated. If SIAS is not actuated under this condition, the operator must proceed to contingency flow and manually actuate SIAS. Step 2 is an ordinary step that checks all of the child instructions and Step 3 checks any one of the child instructions. At any time, the operator can postpone the current step and start the next step. For example, when the operator executes Step 1, he/she can start Step 2 before Step 1 is completed. Moreover, the operator can execute again a step that is already completed. Therefore, CP has 2 types of dynamics: one is the flow of the process and the other is the operator’s workflow changes. Because the operator’s workflow changes are various, modeling with incorporation of these changes is complex. The process of a procedure can be modeled as a flowchart as depicted in Fig. 6, and STPN modeling of the EOP is depicted in Fig. 7.
5. COMPARISON OF COLOURED PETRI NETS MODELING AND STPN MODELING.

5.1 Converting Program of Computerized Procedure to STPN

Converting program of CP to Petri net (CP2PN) has been developed according to the converting rules that are explained in Section 3.2. SKN3.4 computerized procedure was written with eXtensible Markup language (XML), which is readable for both human and machine. The converting program was developed with PIPE (Platform Independent Petri net Editor) that was developed by P. Bonet [8]. CP2PN converts SKN3.4 CP files to Petri net with a format of PNML (Petri Net Markup Language), which was defined by the ISO/IEC 15909 [9,10]. The converted file can be used with various Petri net tools. SKN3.4 operating procedures, including LOCA procedure, has been successfully converted to Petri net by CP2PN.

5.2 Reachability Analysis

Reachability analysis can be used for structural verification of the CP, with operator request for alternative flow. Coloured Petri net modeling of the CP has been used for reachability analysis of normal execution flow. However, additional transitions are needed for alternative execution flow, as explained in Chapter 4. For reachability analysis, the first 2 steps of the LOCA procedure depicted in Fig. 8 have been modeled with a Coloured Petri net. The additional transition for “postpone” was applied. Fig. 9 shows the procedure and the Coloured Petri net modeling. Because in the example CP is very simple, only 2 transitions were added for “postpone”. One is a transition from the first instruction in step 1 (S1I1) to the step 2 starting place (S2S) and the other is a transition from the second instruction in step 1 (S1I2) to the step 2 starting place (S2S). Other alternative execution flows were not considered. The reachability tree from the Coloured Petri net modeling is depicted in Fig. 9. If the postpone is not considered, there are only 8 markings and 1 path in the tree. The postpone part of the modeling in Coloured Petri net adds 12 markings and 11 paths. The STPN modeling from the same procedure is depicted in Fig. 10. The alternative flow of “postpone” has been modeled with the state of the token and the basic flow of the Petri net. The transition between
steps is removed, but each state of the token has a relation with an event that is generated from the token’s state changes. The reachability tree of STPN modeling is depicted in Fig. 10. The STPN modeling has only 8 markings and 3 states of tokens instead of the 20 markings of the Coloured Petri net. Table 2 shows the size of modeling and reachability tree of the Coloured Petri net modeling and STPN modeling. The differences in size will be bigger when the modeling is applied to a whole procedure, which includes all alternative execution flows.

<table>
<thead>
<tr>
<th>Table 2. Size of Modeling and Reachability Tree.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Coloured Petri net Modeling</strong></td>
</tr>
<tr>
<td>Number of Place</td>
</tr>
<tr>
<td>Number of Transition</td>
</tr>
<tr>
<td>Number of Arc</td>
</tr>
<tr>
<td>Number of State of token</td>
</tr>
<tr>
<td>Number of State Space</td>
</tr>
</tbody>
</table>
5.3 Comparison of Complexity of Modeling

For the comparison of the complexity of modeling, 6 steps of the LOCA procedure are used as a sample. It is assumed that the sample procedure is ended at step 6. The sample procedure has been modeled with Coloured Petri net and STPN as depicted in Fig. 11. Fig. 11.B shows the STPN modeling of the sample procedure. The number of

Fig. 11. Modelings of 6 Steps of LOCA Procedure, Including Alternative Execution Flows.
transitions in the STPN modeling is decided by the number and type of instructions at each step. Every instruction has at least one transition. When an instruction has 2 ways of process flow, then it has two transitions. Every step has one additional transition, which is needed for a “start” place. As depicted in Fig. 11, the number of transitions in the STPN modeling is 55. However, Fig 11.A has additional transitions because the Coloured Petri net modeling needs them for the operator’s interrupt. As described in Section 3.3, every instruction needs one transition to the “start” place of next step for “postpone”. Therefore the number of transitions for “postpone” is the sum of the number of instructions at each step, except last. Also, for “change flow”, every instruction needs transitions to all of the steps in the forward direction. The number of transitions for “change flow” in a step is a product of the number of instructions and the number of steps in the forward direction. For “re-execution”, each instruction needs transitions to all steps in the backward direction. The calculating equations for the number of added transitions are provided in Table 3.

The SKN3,4 LOCA procedure has been modeled with the converting program CP2PN that is explained in Section 5.1. It consists of 73 steps, and each step has between one to ten or more instructions. In total, there are 367 instructions. For the alternative execution flow modeling, 26,424 transitions are needed, according to Table 3. The additional transitions and arcs make the modeling complicated, and it is hard to understand normal flow when there are additional transitions and arcs. In STPN those additional transitions and arcs are not needed. The simple state machine in the token can be used to model alternative flow. Also the number of state spaces that are needed for verification is decreased. The complex modeling makes verification, such as reachability analysis, difficult. According to a comparison of the reachability analysis in Section 5.2, reachability analysis of the Coloured Petri net model has markings and paths from each of the added 26,424 transitions.

### Table 3. Required Numbers of Transitions, Arcs and Places for Coloured Petri Net Modeling.

<table>
<thead>
<tr>
<th></th>
<th>Number of Transition (Ntr)</th>
<th>Number of Arc (Na)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Postpone</td>
<td>( Ntr = \sum_{i=1}^{NS-1} N_{li} )</td>
<td>( Na = 2 \times Ntr )</td>
</tr>
<tr>
<td>Change Procedure Flow</td>
<td>( Ntr = \sum_{i=1}^{NS-2} (N_{li} \times (NS - i - 1)) )</td>
<td>( Na = 2 \times Ntr )</td>
</tr>
<tr>
<td>Re-execution</td>
<td>( Ntr = \sum_{i=1}^{NS} (N_{li} \times (i - 1)) )</td>
<td>( Na = 2 \times Ntr )</td>
</tr>
</tbody>
</table>

\( N_{li} = \) Number of instructions in step \( i \)  
\( NS = \) Number of steps in procedure

### 6. CONCLUSION

The Coloured Petri net modeling method for CP has been presented and the converting program (CP2PN) has been developed. Many SKN3,4 CP have been modeled and verified as to the soundness of their structure, such as boundness, liveness and deadlock. The procedure is to be verified in terms of every step and instruction having a valid path of process. The boundness of the procedure flow confirms that procedure flow has a limited possible state. This means that the procedure can be completed within the limited executions of steps.

For the verification of the CP, including an operator’s interruption of procedure execution flow, the STPN modeling method has been presented. The state of a token in STPN can be used for modeling the operator’s actions for procedure execution, including interruptions. For example, when an operator re-executes a step that is already complete, and the result of re-execution is different from previous result, the execution flow of procedure will be changed. In some cases, the already completed part of a procedure is affected by the changed procedure flow. The structural analysis of procedure in this case can be verified with STPN.

When the CPS has STPN as a modeling language for procedure writing and execution, the CPS can provide a structural analysis at the time when the procedure is written. This systematically prevents procedure flow errors and ensures the completeness of the procedure flow. V&V of the emergency procedure also can be supported with this formal verification. Also, qualified and error free procedure execution is essential to procedure based automation, because some controls can affect the safety of the nuclear power plant. STPN modeling and formal verification can be one of the required conditions for procedure-based automation. Also, when CP modeling is applied to a simulator with procedure based automation, many accident scenarios can be simulated and analyzed automatically, and the optimized procedure for the accident can be improved and specified according to the experiment results. Simple expression of STPN also can be applied to other complicated systems that have multi-dynamics, such as interruptions.

### REFERENCES


