NuSCR 정형명세언어를 사용하는 STPA 지원 도구 개발

허윤아 ○ 정세진 유준범
건국대학교 컴퓨터공학과
hyoona1202@naver.com jsjj0728@gmail.com jbyoo@konkuk.ac.kr

NuSTPA 2.0: A Tool to Perform the STPA Using NuSCR Formal Specification

Yoona Heo○ Sejin Jung Junbeom Yoo
Konkuk University, Division of Computer Science and Engineering

Abstract

Systems-Theoretic Process Analysis (STPA) is one of the hazard analysis techniques, which focuses on identifying unsafe controls between components. STPA takes a lot of time and effort to perform, since it is usually done manually by analysts and relies much on their experiences. To perform STPA more efficiently, we proposed a formal approach to support STPA in previous study. This paper proposes a tool to apply STPA process supported by NuSCR formal specification, which follows the proposed formal approach.

1. Introduction

Hazard analysis should be performed in safety-critical systems to identify and mitigate hazards. Systems-Theoretic Process Analysis (STPA) is one of the hazard analysis techniques based on causality model called Systems-Theoretic Accident Model and Processes (STAMP) [1]. It focuses on identifying unsafe controls between components.

However, STPA takes a lot of time and effort to perform, since it is usually done manually by analysts and relies much on their experiences. Therefore, many researchers tried to support STPA by proposing formal approaches and developing tools to perform STPA more efficiently. Like other researches, we proposed a formal approach to support STPA in previous study [2]. And we need to implement a tool to efficiently perform the proposed formal approach.

This paper proposes a tool, *NuSTPA 2.0*, which supports an application of the formal approach proposed in previous studies [2][3]. With *NuSTPA 2.0*, we can apply STPA process supported by Nuclear Software Cost Reduction (NuSCR) formal specification.

2. Background

2.1 NuSCR

NuSCR [4] is a formal software requirements specification method for digital plant protection system in nuclear power plants. It is composed of Function Overview Diagrams (FODs), which represent overview of system components and dependencies to show data flow. It contains nodes called Structured Decision Table (SDT), Finite State Machine (FSM), Timed Transition System (TTS) to represent requirements of a system. Each of these nodes is connected to other related nodes and to related input and output variables. <Fig. 1> shows a <g_LO_SG1_LEVEL> module of NuSCR specification.

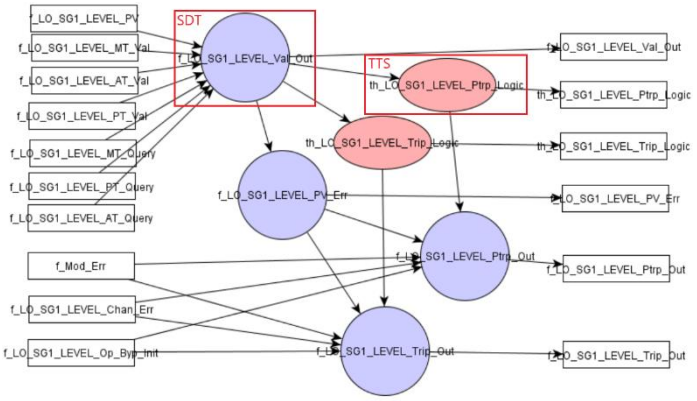

Figure 1. <g_LO_SG1_LEVEL> module of NuSCR specification

NuFTA [3] is a tool to perform fault tree analysis on NuSCR. While analyzing NuSCR, NuFTA draws a fault tree and generates a minimal cut-sets (MCSs). Analysts can set top event for the fault tree and get MCS values.

2.2 Related Work

There have been several tools to support application of STPA. A STAMP tools page at STAMP Workshop site introduces various tools to support STPA. In [5], authors implemented a tool called XSTAMPP (eXtensive STAMP Platform) 2.0, which helps analysts to apply CAST accident analysis and perform extended approach to STPA. Authors in [6] proposes a tool to support STPA process by drawing mind maps to perform STPA step 1 and designing hierarchical control

structures in STPA step 2. Mind maps drawn for STPA step 1 can be later used to analyze traceability. Also, there are more tools introduced in STAMP tools page, such as RM Studio, STAMP Workbench, SafetyHAT, etc.

3. An overview of NuSTPA 2.0

We introduce *NuSTPA 2.0*, which is implemented to apply the formal approach proposed in previous study. A process to be followed by *NuSTPA 2.0* is shown in <Fig. 2>. *NuSTPA 2.0* is developed under JavaFX 11.0.2 and JDK 1.8.0 environment. Requirements that *NuSTPA 2.0* follows are represented in <Table 1>.

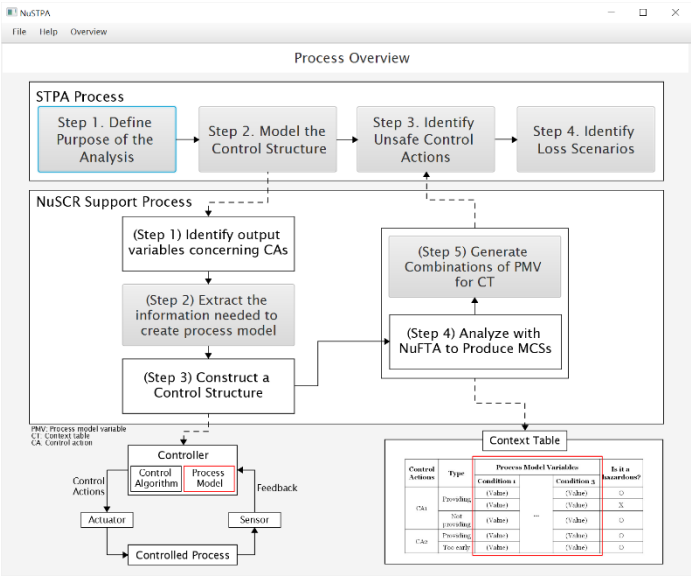

Figure 2. A main screen of NuSTPA 2.0

Table 1. Requirements to be satisfied by *NuSTPA 2.0*

Req. 1	The tool should be able to draw control structure and to show process models in control structure.
Req. 2	The tool should be able to open and parse NuSCR file to automatically obtain process model variables. And it should be able to identify each FODs, variables and SDT, FSM, TTS nodes inside FOD to parse them.
Req. 3	Process model variables and values should be able to be added manually. Generated process model variables should be able to be deleted, modified, and merged.
Req. 4	The tool should be able to open and parse MCS file to automatically obtain context table.
Req. 5	Contexts should be able to be added manually. Items from context table should be able to be modified and deleted.
Req. 6	The tool should be able to automatically generate an unsafe control action (UCA) table from the context table. The UCA table should only show contexts that cause control actions to be unsafe.

The explanation of how *NuSTPA 2.0* satisfies the requirements is as follows:

Req. 1 – *NuSTPA 2.0* provides editor to draw control structure as shown in <Fig. 3>. Process model variables added in NuSCR support process step 2 are automatically added to the control structure.

Req. 2 – *NuSTPA 2.0* uses file chooser and DOM xpath parser to open and parse NuSCR file and follows <Extracting Variable Information> algorithm in [2] to obtain process model variables. The tool contains ArrayLists for each FODs, variables and SDT, FSM, TTS nodes, so that they can be independently handled, and process models can be extracted properly.

Req. 3 – *NuSTPA 2.0* contains manual adding function by entering value into a textField of a popup to add new process model. It also contains deleting, modifying, and merging functions which are enabled with context menu of JavaFX.

Req. 4 – *NuSTPA 2.0* uses file chooser and bufferedReader to open and parse MCS file and obtain contexts to fill context table according to <Generating Context Table> algorithm in [2]. If process model variables are merged, then related context values are also merged into one cell. Each row in context table consists of hazardous contexts which can cause control action to be unsafe.

Req. 5 – *NuSTPA 2.0* enables manual adding of contexts using textField of JavaFX. *NuSTPA 2.0* contains deleting function which is enabled with context menu of JavaFX. Modification of contexts can be done by double clicking the cell to modify the value.

Req. 6 – In STPA step 3, *NuSTPA 2.0* generates UCA table using context table in previous step of NuSCR Support process. Each UCA in UCA table is generated by combining hazardous contexts in each row of context table.

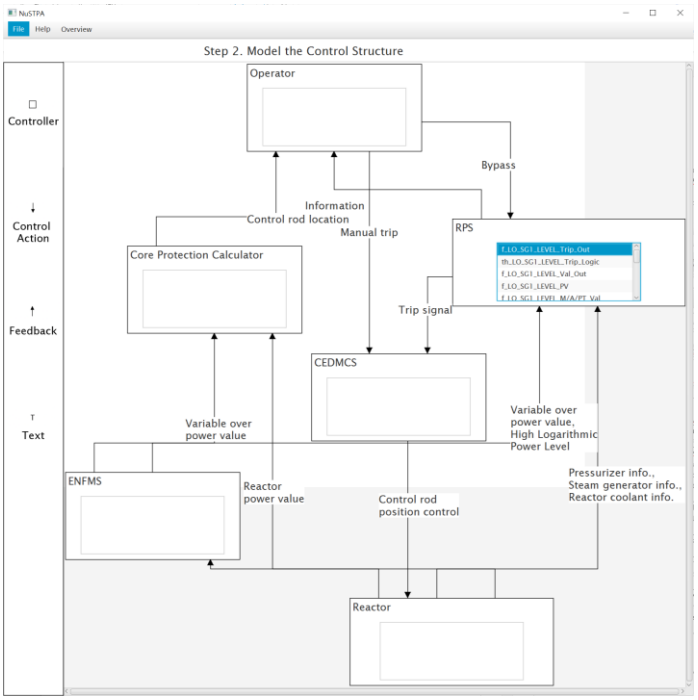

Figure 3. Control Structure of a 'RPS' controller

4. Case Study

We performed a case study on a preliminary version of Reactor Protection System (RPS) Bistable Processor (BP) in a Korean nuclear power plant. We applied same example from previous study [2] to see if the tool works as expected. Nodes inside the module and input variables connected to the module are shown in <Fig. 1>.

<Fig. 4> contains process model variables generated from an output variable <f_LO_SG1_LEVEL_Trip_Out> and connected nodes and input variables which are extracted from selected module <g_LO_SG1_LEVEL>.

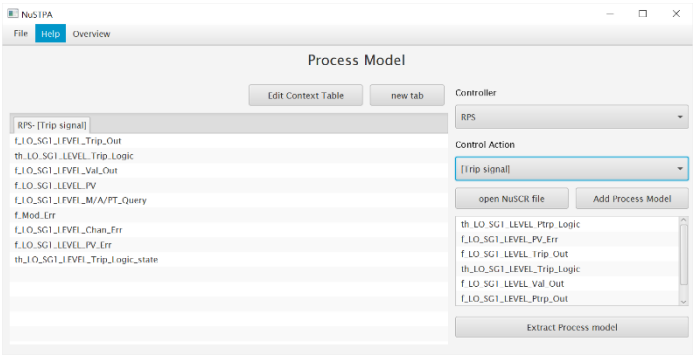

Figure 4. Extracted Process Models for 'RPS' controller

As we follow step 5 in NuSCR support process, we can construct a context table from MCS generated by NuFTA. <Fig. 5> shows a context table automatically extracted from MCS of a selected output variable <f_LO_SG1_LEVEL_Trip_Out>.

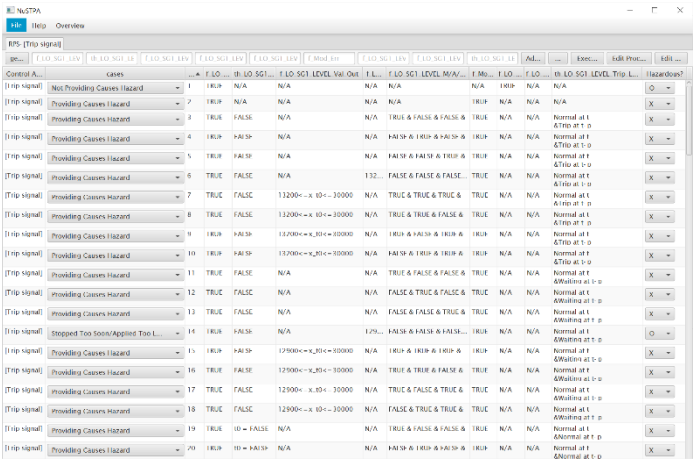

Figure 5. Generated Context Table for 'Trip Signal' CA

As we choose contexts that cause control action to be unsafe, these contexts can be now added to UCA table. An example of a UCA table which has 'Trip Signal' as a control action is shown in <Fig. 6>. These UCAs are using contexts extracted from MCS in previous step and automatically added to UCA table.

5. Conclusion and Future Work

In this paper, we introduced a tool called *NuSTPA 2.0* to apply STPA process supported by NuSCR Formal Specification. It supports

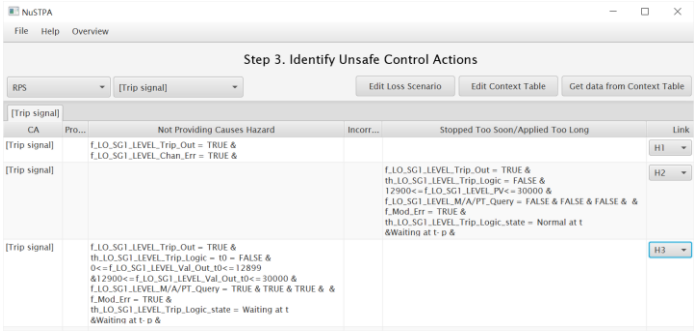

Figure 6. Generated UCA Table for 'Trip Signal' CA

typical STPA process and NuSCR supporting process. It provides editor to draw control structure in STPA step 2 and automatically generates process model by extracting required components from NuSCR specification. Also, it automatically generates context table by extracting components from MCS. Generated contexts can be used to generate UCA table in STPA step 3.

We are planning to improve algorithms to edit control structure, and to generate context tables and UCA tables. We are also planning to enable analyzing system into hierarchical control structure and showing traceability of total process.

Acknowledgements

This paper was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2021R1F1A1047246).

References

[1] N. G. Leveson, "Engineering a safer world: Systems thinking applied to safety", The MIT Press, 2016.

[2] S. Jung, Y. Heo, and J. Yoo, "A Formal Approach to Support the Identification of Unsafe Control Actions of STPA for Nuclear Protection Systems", Nuclear Engineering and Technology (NET), Accepted, 2021.

[3] S. Jung, J. Yoo, and Y. J. Lee, "A software fault tree analysis technique for formal requirement specifications of nuclear reactor protection systems", Reliability Engineering & System Safety, vol. 203, p. 107064, 2020.

[4] J. Yoo, T. Kim, S. Cha, J. S. Lee, and H. S. Son, "A formal software requirements specification method for digital nuclear plant protection systems," Journal of Systems and Software, vol. 74, no. 1, pp. 73–83,2005.

[5] A. Abdulkhaleq and S. Wagner, "XSTAMPP 2.0: new improvements to XSTAMPP including CAST accident analysis and an extended approach to STPA," 2016 STAMP Workshop. MIT, 2016.

[6] S. S. Krauss, M. Rejzek, C. Senn, and C. Hilbes, "SAHRA – an integrated software tool for STPA," in 4th European STAMP Workshop. Zürcher Hochschule für Angewandte Wissenschaften (ZHAW), 2016. [Online].
Available: https://digitalcollection.zhaw.ch/handle/11475/13635