Computerized Procedure Using the Hierarchical State Machine and the Active Object

Sungjin Lee[a*], Yungoo Kim[a], Hyunnam Kim[b]
[a]*Nuclear Engineering & Technology Institute, Korea Hydro & Nuclear Power, 25-1 Jang-dong, Yuseong-gu, Daejeon, 305-343, Korea*
[b]*NWC Consulting, 452, Pyungik Bldg., 990-1, Yeongtong 1-dong, Yeongtong-gu, Suwon-si, Gyeonggi-do, 443-814, Korea*
[*]*Corresponding author: sungjinlee@khnp.co.kr*

1. Introduction

The Advanced Power Reactor 1400 (APR1400) has a Computerized Procedure System (CPS). The CPS is a computerized operator support system that provides operators with graphical user interfaces that contain both plant information and procedural instructions [1]. Nuclear Engineering & Technology Institute (NETEC) has developed CPS applications for the APR1400 dynamic mockup.

In the CPS, one procedure can be executed by operators of different roles and one operator can execute multiple procedures. Each procedure element such as a step or an instruction in a computerized procedure shows execution states whether the element is being executed or completed. Quantum Programming (QP) ™ is one of the widely used frameworks for implementing the hierarchical state machine and the active object [2, 7]. This paper presents that the QP™ framework is an effective means to implement the CPS applications.

2. Implementation Method

2.1 Procedure Execution Model

The dynamic execution flow of a procedure can be represented as shown in Fig. 1. Each transition between two nodes has a direction which means an essential prerequisite on a specific node. Each node means a computerized procedure element such as a procedure, a gross step, a step and an instruction. In addition, each procedure element has its own procedure elements. All nodes and transitions have specific states such as "*NotExecuting*", "*Executing*", "*Completable*" and "*Completed*" in Fig. 2.

Therefore, the hierarchical state machine and the directed graph theory have been adopted for the procedure execution model called "Procedure Execution" in the CPS.

For the multiple procedure execution, the CPS should accommodate up to 20 users and process up to 40 procedures, simultaneously [1]. A crew procedure execution can contain multiple gross steps that should be executed concurrently. Therefore, high concurrency processing capability is necessary in the CPS. For example, in the LOCA (Loss of Coolant Accident) procedure, the operating part can be assigned to shift supervisor and the safety function check part can be assigned to shift technical advisor respectively.

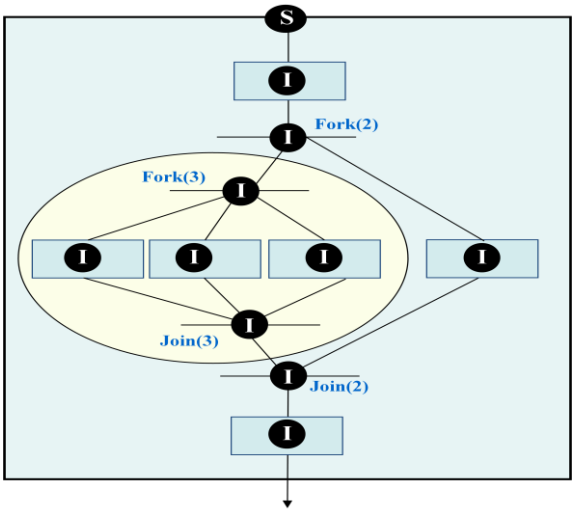

Fig. 1. Hierarchical structure of a step and its associated instructions in a computerized procedure execution

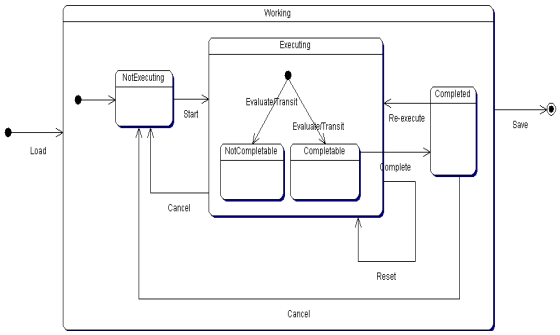

Fig. 2. A state chart of a procedure element

2.2 Procedure Execution Handling

When a procedure is executed in the CPS, various signals and events such as the plant parameter updates and operator interactions should be processed on the computerized procedure, simultaneously regardless of their sequences. This functionality can be generally provided by the multithreaded programming technique. But there are some generic problems when the multithreaded programming is adopted. One of the most serious problems is to ensure mutually-exclusive

operations in order to prevent common data from being simultaneously modified or read while multiple threads are processed concurrently. In addition, it is important to prevent deadlocks caused by careless use of such primitives [3].

An active object decouples function execution from function invocation for each thread. This decoupling is achieved through asynchronous invocation and event handling by the active object [4]. QP™, one of the active object frameworks, has been adopted to transfer correct events into a "Procedure Execution".

2.3 Procedures Concurrency Handling

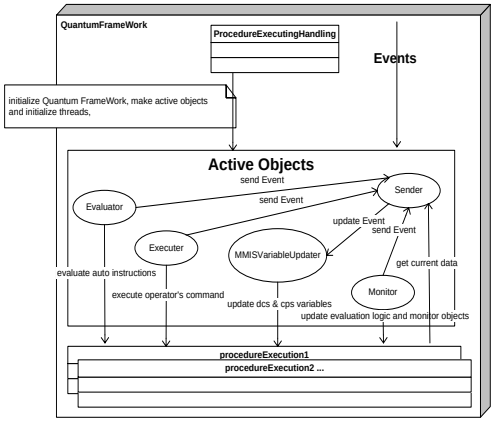

Fig. 3. Operations among state machines and active objects in the CPS application

In the CPS implementation, there are active objects such as "Evaluator", "Executer", "Updater", "Monitor" and "Sender" as shown in the upper part of Fig. 3. Hierarchical state machines representing procedure (i.e. procedure execution) are created for each procedures used as shown in the lower part of Fig. 3. All the active objects and state machines are resides in a common space where messages among all objects are communicated.

3. Development Result of the CPS Applications

The CPS has been developed as shown in Fig. 4. The CPS consists of server and client applications based on Qt [5, 6]. They were installed and used in the dynamic mockup for the preliminary HFE verification and validation of ShinKori Unit 3&4. One low performance server application and five client applications were connected via network in the dynamic mockup.

Any request from an operator client is transferred to the server and its processing result is sent back to the operator client. Finally, the client application shows the result to the operator. In this process, the response time on the operator's request is met in the requirements' criteria. In addition, high performance computer hardware was not necessary to run the CPS application.

By introducing the framework, software development complexity of the CPS application became much low and this low complexity resulted in low risks in the violation of mutually-exclusive operations, deadlocks and loss of events.

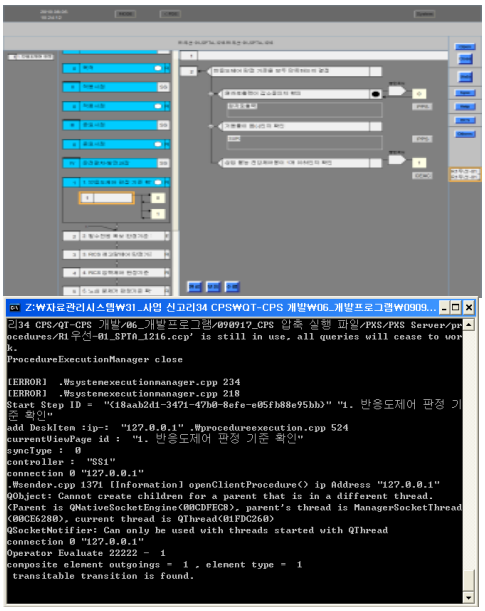

Fig. 4. Screen shots of CPS client (top) and server (bottom) applications

4. Conclusions

The implementation of the CPS was successful and used for the preliminary HFE verification and validation in the APR1400 dynamic mockup. By applying the well-proven programming framework, the computer hardware performance and the software development complexity were lowered considerably while the performance requirement for the response time and number of users was met. It is recommended that the use of the hierarchical state machine and the active object framework should be an effective approach to implement a real-time software application that has composite states and requires highly concurrent processing.

REFERENCES

[1] System Specification for Computerized Procedure System (DDS1), Korea Hydro & Nuclear Power, 2009.
[2] D, Schmidt; M., Rohnert, H., Buschmann, F., Pattern-Oriented Software Architecture, Volume 2: Patterns for Concurrent and Networked Objects, John Wiley & Sons, 2000.
[3] http://en.wikipedia.org/wiki/Thread_(computer_science).
[4] Bass, L., Clements, P., Kazman, R. Software Architecture in Practice. Addison Wesley, 2003.
[5] http://doc.qt.nokia.com/4.6/gpl.html.
[6] http://qt.nokia.com/products.
[7] Miro Samek, Practical UML Statecharts in C/C++, 2002.