Graph Database Design of Control Logic Drawings

Dongil Lee [a*], Donghun LEE [b]
[a]*KHNP, Central Research Institute, 70, 1312-gil, Yuseong-daero, Yuseong-gu, Daejeon, 34101, South Korea*
[b]*HUPEC. Co., Ltd., #1810, 10, Dongtan-daero 21-gil, Hwaseong-si, Gyeonggi-do, 18471, South Korea*
[*]*Corresponding author: diturtle@khnp.co.kr*

1. Introduction

Control logic drawings used to control Nuclear Power Plants (NPPs) exist in various forms, such as hardcopy and files (pdf, jpg, tif files, etc.). It was necessary to understand the meaning of symbols and understand the flow of signals along lines for verification of drawings.

However, there are clearly elements of uncertainty in human work. Uncertainty causes logic errors, which can have catastrophic consequences for the plant.

To reduce fatal mistakes, verified procedures and validated results through software Verification and Verification (V&V). So far, verification has been done manually.

We have been carrying out the task since April 2020 to verify various types of control logic drawings used in nuclear power plants. In this project, Control Logic Diagram (CLD) and Vendor Logic Diagram (VLD) are recognized based on Artificial Intelligence (AI) technology and reproduced as normal vector drawings (Auto CAD format), and normal vector drawings are tested and verified [1].

When a drawing is recognized and a normal vector drawing is produced, the drawing and all symbols, lines, and characters in the drawing should be databased for the test of the drawing.

The most used database is the Relationship DataBase (RDB), but it requires a very large database and a large amount of computation to express many symbols, lines, characters, and relationships. To overcome this limitation, we try to exceed the limitation of RDB by expressing the drawing using the Graph Database [2].

In addition, compatibility was maintained by using eXtensible Markup Language (XML) for Graph Database for compatibility with widely used RDB.

2. Why Neo4j

In this section, Graph Database Neo4j used to express drawings and useful for expressing control logic drawings are explained.

2.1 Use of Graph Database Neo4j

RDBs have been and will continue to be used in many fields and industries to date. However, there is a lot of loads due to the increasing amount of data and operations such as joins in query statements. Efforts are being made in many fields to solve it, and it has been widely used even with a lot of loads.

For example, if a result is to be found through several different symbols and drawings in an input signal, RDB must go through numerous joins such as symbols, drawings, and connection tables. In addition, since the same symbols exist in the drawings, it is difficult to normalize the DB. However, in the graph DB, there is no join operation, only Relationships are checked, and unnecessary operations and data are not referenced.

In this project, changes in drawings must be reflected in real time, so the computational load of the DB should not be increased by joining tables and searching for unnecessary data.

2.2 Data Modeling in Neo4j

Neo4j has four (4) basic data structures: Nodes, Relationships, Labels, and Properties. (1) Nodes means general object information. (2) Relationships are like the Join command in RDB and have a type, a start node, an end node, and a direction. (3) Properties has properties (name/value) of Node and Relationship. (4) Labels are a method of classifying nodes [3].

2.3 Cyper

Structured Query Language (SQL) used in RDB uses explicit query language, but Neo4j uses Cyper, a declarative query language that even beginners can use easily.

In order to obtain the desired data from the RDB, it is necessary to understand the structure of the database (structure, relationship of the table). However, to obtain the data desired by the user by using the declarative query language, it is easy to find it without understanding all the structure of the database.

3. Construct control logic drawing with Neo4j

3.1 Neo4j Database Configuration

Database configuration is defined to include “project work area, drawings, drawing settings, symbols, symbol settings, symbol-symbol association, and drawing association”. (See Fig. 1.)

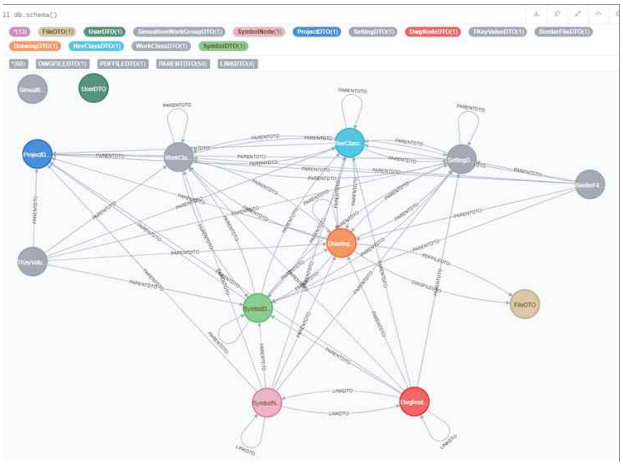

Fig. 1. Neo4j based database configuration

Table I: Database Class

Class	Description
Project	Project properties
WorkClass	Work area of each project
RevClass	Revision division
Setting	Drawing and symbol setting by Rev.
border	Drawing files belonging to Settings
Drawing	RevClass, sub-drawing information
DwgNode	Node connected to Drawing
Symbol	Symbols and connected to Drawing
SymbolNode	Nodes connected to Symbols
ConnectionLine	Nodes to nodes connection (Between SymbolNode, DwgNode)
User	User information
Simulation	Simulation information

3.2 Neo4j Configuration of Control Logic Drawing

To compose a control logic drawing in Neo4j, it consists of "drawing information, symbol information, symbol properties, drawing connection information, drawing symbol connection information, and symbol connection information". (See Fig. 2.)

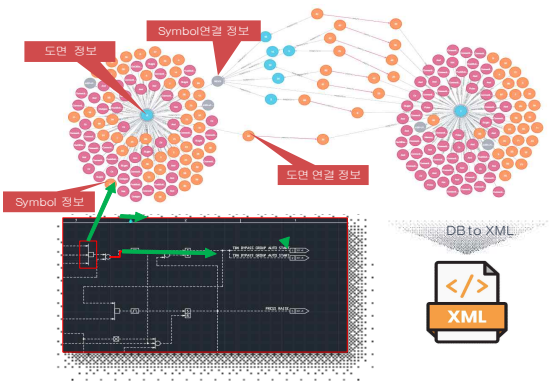

Fig. 2. Drawing and Neo4j connection structure

3.3 Database Compatibility using XML

The purpose of XML in the implemented database is to convert data, store objects, and write internal data of programs.

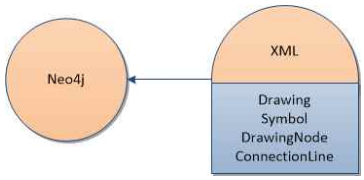

Fig. 3. Storing drawing information in XML

Table I: XML Elements

Elements	Description
DrawingDTO	Drawing information
SymbolDTO	Symbol configuration
DrawingNodeDTO	Node connected to drawing
ConnectionLineDTO	Nodes to nodes connection

4. Conclusions

To test control logic drawings of general hardcopy and files (pdf, tig, jpg, etc.), it is essential to configure a database for computerization of control logic drawings.

This is because the logic must work by giving life to a dead drawing. For the control logic drawing to come alive, it is essential to configure an appropriate database for the control logic drawing.

In one drawing, there are as few as thirty (30) symbols and as many as thousands of symbols, and the drawings are linked from one to several thousand.

When a control logic drawing with direction, properties, and many connections is implemented with a RDB, the load increases when the amount of data of calculations and symbols increases. Therefore, an appropriate graphic database should be applied.

REFERENCES

[1] D. I. LEE, Development of Digital Control Logic's Verification Technology based on Artificial Intelligence, Conference on Information and Control Systems, p397-398 ,2020

[2] Miller, Justin J., Graph database applications and concepts with Neo4j, Proceedings of the southern association for information systems conference, Atlanta, GA, USA. Vol. 2324. No. 36. 2013.

[3] Baton, Jerome, Learning Neo4j 3.x - Second Edition: Effective data modeling, performance tuning and data visualizat, chapter 3, 2017