Formation of an information retrieval system of visual programming scenarios for information modeling software complexes

Angelina Rybakova* and Daniil Piatkov
Moscow State University of Civil Engineering, 26, Yaroslavskoye shosse, Moscow, 129337, Russia

Abstract. This article discusses the process of creating an application based on a specialized database for working with a variety of scripts – visual programming scenarios. The SQL query Language (Structured Query Language) was used as a key tool for implementing the application, data for systematization was presented from the file system for storing scripts of the organization in question. In view of the popularization of the use of visual programming tools in information modeling software complexes, there is a problem of systematization of a regularly increasing number of scripts both in a specific organization and in the market as a whole. The purpose of this work is to develop an information retrieval system for working with visual programming scripts, the tasks are to analyze the file structure of script storage, form database requirements, develop algorithms for database formation and dynamic SQL query formation to filter the necessary scripts. Automation of the implementation of all the above tasks is performed on the basis of the high-level programming language C++ with the integration of SQL queries in the cross-platform development environment Qt Creator. As a result, an information search engine is presented in the form of an application "Script Manager".

1 Introduction

In recent years, visual programming tools in the information modeling environment have become increasingly popular. Visual programming is a way of developing a program based on manipulations with graphic objects instead of writing code in the form of text. This tool allows you to generate script code even without knowledge of the syntax of the programming language. [1-2]

The main purpose of using visual programming, including in the information modeling environment, is to expand the functionality of the software package by developing its own functions. As a result, designers can automate routine tasks and go beyond the basic functionality of the program.[3]
Visual programming includes a library of previously prepared program code blocks in the form of a graphical node element. Formed in a certain way by a set of nodes is a script—script, as a result of which one or another task is performed automatically. Simultaneously with the basic node library, additional packages created by users can be added to the program.[4-5]

The spread and development of visual programming in the information modeling environment has led to a sharp increase in the number of scripts both in a particular company and in the industry as a whole. And already today, large project companies are faced with the difficulties of interacting with a large number of scripts that perform various functions. There are problems with storing, searching, processing and updating scripts, which leads to an increase in the duration of the design and additional labor costs of specialists. [6-7]

Thus, it is advisable to create a tool that will eliminate the problems of working with a large number of different types of scripts. The purpose of this work is to develop an information retrieval system for working with visual programming scripts, the tasks are to analyze the file structure of script storage, form database requirements, develop algorithms for database formation and dynamic SQL query formation (Structured Query Language) to filter the necessary scripts.

2 Metods

2.1 File structure of script storage

Taking into account the design experience of the project company in question, a certain file system was formed during the work. The structure of this file system is based on the key sections of the project documentation. As a result, script storage is determined by four functional purpose groups within the section: architectural and design solutions, engineering support and regardless of the section. The further hierarchy is formed on the basis of combining the customer's objects with subsequent detailing for each specific object. Each script is located in accordance with the address described above according to the section of the project documentation, the customer, the object and the marking. Fig.1.

![Fig. 1. File structure of script storage.](image-url)
The final path of the script address may include several different scripts for the same hierarchy groups, but different solved functions or tasks. In this case, duplication is eliminated due to the corresponding unique file name. [8-10]

After analyzing this script storage system, the next task is to integrate it into the database.

2.2 Database Requirements

To systematize scripts in the file system, it is necessary to determine the universal characteristics of scripts inherent in each, but different from each other. Taking into account the specifics of information modeling technologies, the features of the corresponding design tasks are added to the set of script characteristics. [11-12] Thus, it is possible to determine the following features of the scripts that form the basis of the database table:

1. The location of the script in the file structure.
2. The name of the script.
3. A brief description of the functionality.
5. Name of the customer.
6. The construction object.
7. Section of project documentation.
8. Identification tag.

Taking into account the specifics of the database construction, the above features are detailed and grouped into two tables. (Fig.2) The schematic diagram of the database-based application is shown in Figure 3.

![Fig. 2. Database table.](image-url)
On the basis of the presented system, storage with elements of script systematization will be carried out. Also, taking into account the structure of this database, a software implementation of the storage system is being developed directly.

2.3 Database formation algorithm

The main difficulty of creating a database of scripts for further integration into the application is to manually enter the characteristics of scripts for technical filling of the library. [13-15] Given the large number of scripts in the company, this procedure seems to be the most time-consuming and time-consuming. Therefore, in order to rationalize this stage, it is advisable to develop an automated method for creating a database.

Automation of filling in database tables, along with minimal manual input, consisted of the following methods:
1. Scanning script markup using RegExp regular expressions.
2. Formation of tags based on the location of files.
3. Transfer of labels to tags.

Conceptually, the formation algorithm consists of searching, defining and transferring script attributes to the database storage system, discussed in Section 2.2. In case of collisions or lack of data, expansion of the database structure, SQL query logic or the use of manual input is provided. Verification and validation of the database is performed by the corresponding SQL query. [16] If there is a lack of a particular attribute, a manual adjustment is performed.

2.4 Algorithm for dynamic formation of an SQL query

SQL (Structured Query Language) is a query language for working with databases structured in a special way. The task of SQL is to compose queries in such a way as to find among a large amount of information that is needed for specific purposes, sort it, structure it and present it in the simplest and most understandable way. An SQL query is a set of commands for working with tabular databases. [17-18]
Using SQL to work with a script database makes it possible to easily access a large number of scripts without additional labor. Based on a number of filters and a search string, an SQL query is generated to find the required script. This is how filtering a large amount of data works. [19-20] The information search engine includes the following fundamental rules of query formation:

- if the filter value is not equal to "Off", then AND column='value' is added to the SQL query;
- if the "search incl. in description" button is pressed with other active filters, the search string is added to the query as AND (name='%value%' OR description='%value%');
- if all filters are disabled and the search string is empty, then WHERE will not be in the query. The entire list of existing scripts will be displayed;
- if only string search is enabled without other filters, the query looks like this ...WHERE name='%value%' OR description='%value%'.

Automation of the implementation of all the above tasks is performed on the basis of the high-level programming language C++ with the integration of SQL queries in the cross-platform development environment Qt Creator.[21]

3 Results

Based on the above data and methods, the Script Manager application was developed (Fig. 4) using SQLite databases and a user interface. The application is a tool for accessing a variety of scripts in order to quickly find the most suitable one within the framework of the project task being solved by a specific specialist.

![Fig. 4. Program interface.](image-url)

For the most convenient search for the required script, a set of filters was implemented:

- version (the version of the software package, in this case Autodesk Revit, for which the script is intended);
- customer;
- object;
Due to the high speed of SQL query execution, script search and filtering are performed instantly. Before using the script, it is necessary to study its instructions. In the standard mode, the execution of this process is quite time-consuming, since the description of the script is available only when the script itself is directly opened in the program. This problem is solved by adding instructions in the right part of the application interface (without the need to tear off the script). This makes it much faster to determine the purpose of the script and find the necessary one.

To open the script, Drag and Drop technology was implemented: in order to use the script, it is enough to drag a line with information about the script into the Dynamo interface (visual programming interface built into Autodesk Revit. Previously, the opening was possible only through the Dynamo interface.

In order to analyze the popularity of the application, it was decided to log (the process of recording information about events), user actions, such as applying filters and opening scripts. Fig. 5.

Since a large number of scripts have been developed in the organization in question, it is quite time-consuming to enter this information into the database manually. Thus, it became necessary to automatically read and fill in information about scripts. As part of this task, an algorithm was developed that recursively scans the storage location of script files in the organization and fills in all the necessary information based on the name of the script, the path along which it is located, as well as by searching for comments in the source code of the script using regular expressions (for example, instructions / description of the script). The automatic filling operation takes 28-30 seconds.
4 Discussion and Conclusions

Thus, a mechanism for accounting and searching for scripts has been developed, which makes it easy and simple to find a new script, without having to manually search through the file structure, open scripts to read the description and search for what is needed. The mechanism also avoids the problem of collisions and re-writing scripts of the same functionality. The application increases the frequency of using scripts, as it minimizes situations when a specialist cannot find the necessary in a haphazard storage structure and, as a result, do previously automated work manually.

Taking into account the frequency of use of the script and the time of its operation, it becomes possible to analyze the economic effect of the development of each script. As a result, this provides a basis for assessing the feasibility of applying and making design and management decisions both within the framework of a single project and in the work of the company. The evaluation data is generated based on logging.

Taking into account the feedback from the company's leading experts on testing the prototype of the application, suggestions for improvement and further development are formulated. What ultimately determines the prospects of this development and new directions for work and research.

References

1. G. Nocerino, M. F. Leone, Energy and Buildings 278, 112626 (2023)
5. X. Zhao, Y. Jin, N. M. Selvaraj, M. Ilyas, C. C. Cheah, Automation in Construction 154, 104996 (2023)
6. Y. Ding, M. Liu, X. Luo, Automation in Construction, 144, 104580 (2022)
7. Y. G. Wang, X. J. He, J. He, C. Fan, Automation in Construction, 141, 104395 (2022)
15. Y. Teng, J. Xu, W. Pan, Y. Zhang, Building and Environment, 221, 109260 (2022)

18. M. M. N. Abdelhady, M. M. Karmanova, Components of scientific and technological progress, **7(73)**, 32-37 (2022)

