Library Monitor: Enhancing C/C++ Development Efficiency and Stability

Chieh Huang
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

Chen Tse, Chen
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—**
***Index Terms*—**

I. Introduction

A. Problem statement

In the process of software development, using libraries is a common practice. Nowadays, there are also many package managers developed specifically for certain programming languages, such as PyPI for Python, Maven for Java and Kotlin. However, there is currently no designated package manager for C/C++. One challenge that developers may face is the updates and modifications to libraries, where functions used in the code may become obsolete or generate errors. To address this problem in C/C++, we propose a project called 'Library Monitor'.

B. Motivation

In C/C++, as library versions update, developers need to constantly keep track of these changes to ensure smooth operation of their code. This requires a significant amount of time and effort, and there's a chance of errors. Furthermore, when functions within libraries are modified or removed, code that relies on these functions may be affected, leading to program errors. These issues not only impact development efficiency but also diminish software quality and maintainability.

C. Relation with software engineering research

In the field of software engineering, ensuring the stability and maintainability of code is crucial. This includes adapting to updates and modifications in libraries to ensure compatibility with new versions. Modifying code according to the content of new versions to avoid errors and compatibility issues. However, there are still some unknowns, such as how to effectively monitor changes in libraries, and how to provide developers with real-time updates and suggestions.

D. Key Insight or Idea

This system would track changes in libraries, including updates, modifications, and deprecated functions, and provide real-time notifications to developers. It would also analyze code dependencies and suggest necessary updates or modifications to maintain compatibility with new library versions. This approach aims to enhance development efficiency, improve software quality, and ensure the stability and maintainability of code in the ever-evolving landscape of software engineering.

E. Assumptions

1. Libraries are publicly available and structured, allowing the system to automatically analyze them.

2. Documentation is provided for library updates and modifications, allowing visibility into version trends.

3. Machine learning and natural language processing techniques can accurately analyze library update logs and code changes, extracting key information.

4. Developers will use the library monitoring system we provide.

F. Research questions

1. How to design a system to monitor library changes and provide developers with real-time update information?

2. How can machine learning and natural language processing techniques be applied to analyze library updates?

G. Evaluation Dataset

We can obtain the updates and code changes of libraries from open-source code repositories such as GitHub, Bitbucket, etc., which are often publicly available.

H. Evaluation metrics

Tracking each library's history changes and comparing the differences between the historical versions and the current release is an essential aspect of solving the problem successfully.

II. Background and Motivation

III. Related Work

Package Managers: Package managers such as NuGet, CocoaPods, and Composer manage software packages and libraries, handling updates, dependencies, and versioning. They provide a centralized repository for developers to discover, install, and manage libraries.