ХЕРСОНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
(повне найменування вищого навчального закладу)
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ
(повне найменування інституту, назва факультету (відділення))
КАФЕДРА ПРОГРАМНИХ ЗАСОБІВ ТА ТЕХНОЛОГІЙ
(повна назва кафедри (предметної, циклової комісії))

Пояснювальна записка

до кваліфікаційної роботи
бакалавра
(освітній рівень)

на тему: «Розробка програмних засобів аналізу статистичного аналізу програмного коду»

Виконав: студент 4 курсу, групи 4ПР2
спеціальності
121 - «Інженерія програмного забезпечення»
(шифр і назва спеціальності)

Мойсеєнко Амет Аметович
(прізвище та ініціали)
Керівник д.т.н., проф. Шерстюк В.Г.
(прізвище та ініціали)
Рецензент _____
(прізвище та ініціали)

Херсон - 2021

Херсонський національний технічний університет
(повне найменування вищого навчального закладу)

Факультет, відділення Інформаційних технологій та дизайну
Кафедра Програмних засобів і технологій
Освітній рівень бакалавр
Спеціальність 121 – Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри
Програмних засобів та технологій
д.т.н. проф. В.Г. Шерстюк
"___" _____ 2021 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Мойсеєнко Амету Аметовичу
(прізвище, ім'я, по батькові)

Тема роботи «Розробка програмних засобів аналізу статистичного аналізу програмного коду»

керівник роботи д.т.н. професор Шерстюк В.Г.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджена наказом вищого навчального закладу від . .2021 р. № -

2. Строк подання студентом роботи _____

3. Вихідні дані до роботи літературні та періодичні джерела, матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):

1) опис предметної області аналізу програмного коду; 2) теоретичні основи аналізу програмного коду; 3) аналіз методів статичного аналізу коду; 4) опис програмного продукту; 5) модифікований алгоритм статичного аналізу

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

1) діаграма варіантів; 2) побудова AST дерева; 3) розбір дерева за методом Хіндлі-Мілнера; 4) архітектура програмного забезпечення; 5) серверна архітектура на основі Fastify; 6) діаграма компонентів; 7) результати експериментів.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання роботи	Термін виконання етапів роботи	Примітки
1.	Отримання завдання	08.02.2021	Виконано
2.	Підбір літератури	09.02.2021-21.02.2021	Виконано
3.	Аналіз предметної області	22.02.2021-28.02.2021	Виконано
4.	Розробка та обґрунтування завдання	01.03.2021-14.03.2021	Виконано
5.	Розробка концептуальної моделі	15.03.2021-21.03.2021	Виконано
6.	Розробка алгоритму	22.03.2021-04.04.2021	Виконано
7.	Проектування програної системи	05.04.2021-11.04.2021	Виконано
8.	Розробка інтерфейсу програмної системи	12.04.2021-18.03.2021	Виконано
9.	Тестування програмної системи	19.04.2021-25.04.2021	Виконано
10.	Оформлення пояснювальної записки	26.04.2021-09.05.2021	Виконано
11.	Захист кваліфікаційної роботи	15.06.2021	Виконано

Студент _____ Мойсеєнко А.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Шерстюк В.Г.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 99 сторінок, 22 рисунка, 5 таблиць, 1 додаток, 20 джерел.

Мета роботи – покращення процесу аналізу вихідного коду, розробка алгоритму, що виконує більш глибокий типізований аналіз програмного коду.

Об’єкт дослідження – процес статичного аналізу вихідного коду програм.

Предмет дослідження – алгоритми статичного аналізу синтаксичної та семантичної моделі програмного коду.

Результати роботи:

− досліджено методи статичної обробки та аналізу коду;

− розроблено алгоритм, що забезпечує збільшення інформації про код в процесі запуску;

− виконано програмну реалізацію алгоритму.

Новизна роботи: полягає у отриманні методу аналізу статичної та семантичної моделі коду на надійність на основі модифікованого алгоритму виводу типів Хіндлі-Мілнера, що забезпечує отримання більш детальної інформації про стан вихідного коду та дозволяє підвищити його якість.

Практичне значення одержаних результатів полягає в тому, що розроблений спосіб верифікації програмного забезпечення надає більш детальну інформацію щодо якості вихідного коду, наприклад, об'єктів, що мають змінюваний стан та є поліморфними та може використовуватись в процесі розробоки програмного забезпечення в якості верифікатора та в процессі в навчать в якості контролю.

Ключові слова: *статичний аналіз коду, шаблони проектування, потік даних, функціональне програмування*.

АНОТАЦІЯ

Кваліфікаційна робота бакалавра складається зі вступу, п’яти розділів, висновку, списку використаних джерел та додатку.

При виконанні кваліфікаційної роботи вирішувалась задача удосконалення статичного аналізу тексту програм, а саме аналізу потоку даних. Як визначено в першому розділі, проблематика полягає в тому, що існуючі методи не можуть давати достовірні результати при інкапсульованому підході до побудови об'єктів, а також при використанні поліморфізму.

В другому розділі було розглянуто вже відомі алгоритми статичного аналізу, кожен з яких має як переваги так і недоліки, які були розглянуті та описані. Ні один з існуючих алгоритмів не дає рішення задачі статичного аналізу у тому обсязі, у якому його вирішує синтезований алгоритм.

В третьому розділі запропоновано алгоритм, що є модифікацією алгоритму data-flow аналізу, який полягає в зміні структури даних для кожної інструкції, що дозволяє обробляти код з інкапсульованими та поліморфними об’єктами.

В четвертому розділі розроблено метод аналізу програмного коду, який складається з елементів аналізу потоку даних та алгоритму виводу типів Хіндлі-Мілнера. В якості вхідних параметрів він приймає програмний код на мові C#, проте, за своєю структурою, є універсальним і придатний до застосування для інших мов із C-подібний синтаксисом та з об'єктно-орієнтованими можливостями.

П’ятий розділ присвячено програмній реалізації розроблених алгоритму та методу статистичного аналізу програмного коду.

ABSTRACT

The bachelor's thesis consists of an introduction, five sections, a conclusion, a list of sources used and an appendix.

When performing the work, the problem of improving the static analysis of the program text, namely the analysis of the data flow, was solved. As defined in the first section, the problem is that existing methods cannot give reliable results with an encapsulated approach to the construction of objects, as well as when using polymorphism.

In the second section, the already known algorithms of static analysis were considered, each of which has both advantages and disadvantages, which were considered and described. None of the existing algorithms provides a solution to the problem of static analysis to the extent that it is solved by the synthesized algorithm.

The third section proposes an algorithm that is a modification of the data-flow analysis algorithm, which is to change the data structure for each instruction, which allows you to process code with encapsulated and polymorphic objects.

In the fourth section, a method of software code analysis is developed, which consists of elements of data flow analysis and Hindley-Milner type output algorithm. As input parameters, it accepts program code in C #, however, by its structure, is universal and suitable for use in other languages with C-syntax and object-oriented capabilities.

The fifth section is devoted to the software implementation of the developed algorithm and method of statistical analysis of program code.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AST – abstract syntax tree.

API – application programming interface.

JSON – JavaScript Object Notation.

SOLID – набір з 5 правил, що повинні виконуватись будь яким розробником при кодуванні.

REST – Representational State Transfer, система правил для побудови API

SQL – мова запитів до реляційної бази даних.

SQL-ін’єкція – метод атаки на реляційну базу даних.

Statefull-об’єкті – об’єкти, дії яких залежать від поточного стану, який є змінюваним.

ООП – об’єктно-орієнтоване програмування.

CDG – граф залежних умов.

CFG – граф потоку управління.

MVC – паттерн на основі компонентів model(модель), view(представлення), controller(контроллер).

ВСТУП

Етап верифікації програмного забезпечення включає в себе складну систему обробки вихідного коду та сможе бути виконаний великою кількістю способів – це і синтаксичний і семантичний аналіз, також перевірка на наявність неактивних блоків коду, умов, що ніколи не виконується, повторення, помилки орфографії. Цій темі посвячена невелика кількість наукових робіт, оскільки ця тема сама по собі є допоміжною в розробці, а не виконання окремих функціональних блоків.

Існуючі накові роботи впроваджують процеси перевірки безпеки, зокрема використання API, SQL-ін’єкції, пошук паттернів, і тд. Проте наразі існує досить невелика кількість робіт стосовно аналізу потоку вихідного коду [1-5]. Одним із невирішених питань є аналіз поліморфного коду, а також коду, що пов’язаний зі змінюваним станом об’єктів. Тому одною з головних цілей цієї роботи є подолання цього бар’єру для отримання більш якісної картини по коду а також перевірка можливості використання алгоритмів виводу типів з функціональних мов програмування до імперативного коду.

Другорядними елементами роботи є аналіз рис програмного коду, зокрема цикломатичної складності та пошук використаних шаблонів програмування. Разом цей функціонал складає комплексну систему верифікації та виведення рис з програмного коду, та по своїй суті складає невирішену проблему повного статичного аналізу, проте, з іншої сторони є лише частковим підходом в системі в обробки вихідного коду.

Дане дослідження і розробка алгоритмічного забезпечення є актуальним для команд інженерної розробки та в навчальному процесі. Потреба в більш якісному виконання етапу верифікації зростає, особоливо це помітно в нових галузях, що пов’язані з прийняттям рішень та систем реального часу, а отже і зростають запити до систем статичного аналізу.