Міністерство освіти і науки України

Запорізький національний технічний університет

кафедра програмних засобів

**РЕФЕРАТ**

З дисципліни «Якість програмного забезпечення та тестування» з теми:

**«Статичний аналіз як метод верифікації програмного забезпечення.**

**Перевірка правил коректності»**

Виконав:

студент групи КНТ-115 Ю.О. Лукашенко

Прийняла:

професор, д.т.н Г.В.Табунщик

2018

# ЗМІСТ

[ЗМІСТ 2](#_Toc528315217)

[ВСТУП 3](#_Toc528315218)

[1 ВЕРИФІКАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 4](#_Toc528315219)

[1.1 Загальні відомості 4](#_Toc528315220)

[1.2 Огляд методів верифікації 4](#_Toc528315221)

[1.3 Статичний аналіз програмного забезпечення 6](#_Toc528315222)

[1.4 Перевірка правил коректності 7](#_Toc528315223)

# ВСТУП

Однією з найважливіших проблем при розробці, розробці програмного забезпечення (ПЗ) і тестуванні є його верифікація. Методи перевірки програмного забезпечення, призначені для підтвердження фактів відповідності кінцевого програмного продукту вимогам заявника, тож для перевірки програмного забезпечення це виявлення помилок, вразливостей, некоректно реалізованих властивостей та вимог[1,2]. Проблема створення нової класифікації методів перевірки ПЗ актуальна, оскільки дозволяє розглянути існуючі на даний момент методи перевірки ПЗ та їх програмну реалізацію, виявити їх переваги та недоліки. Існуючі методи перевірки ПЗ можна розділити таким чином:

* емпіричні – такі, що використовують експертизу;
* формальні – такі, що використовують математичний апарат для верифікації програмного забезпечення;
* статичні – такі, що перевіряють роботу програмної системи без її запуску;
* динамічні – такі, що перевіряють роботу програмної реалізації за допомогою безпосереднього запуску[3].

З точки зору рівня автоматизації, методи перевірки ПЗ можна розділити на ручні, автоматизовані та автоматичні.

# 1 ВЕРИФІКАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні відомості

Верифікація програмного забезпечення є комплексним поняттям, що включає у себе підтвердження коректності окремих об’єктів програми або програмної системи в цілому. Метою верифікації є досягнення гарантії того, що верифікований об'єкт відповідає вимогам, реалізований без непередбачених функцій і задовольняє проектним специфікаціям і стандартам[4]. Процес верифікації включає в себе інспекції, тестування коду, аналіз результатів тестування, формування та аналіз звітів про проблеми. Таким чином, прийнято вважати, що процес тестування є складовою частиною процесу верифікації.

Загальноприйнятий поділ методів верифікації [1,2,4] можна представити в вигляді діаграми, зображеної на рис. 1.1.

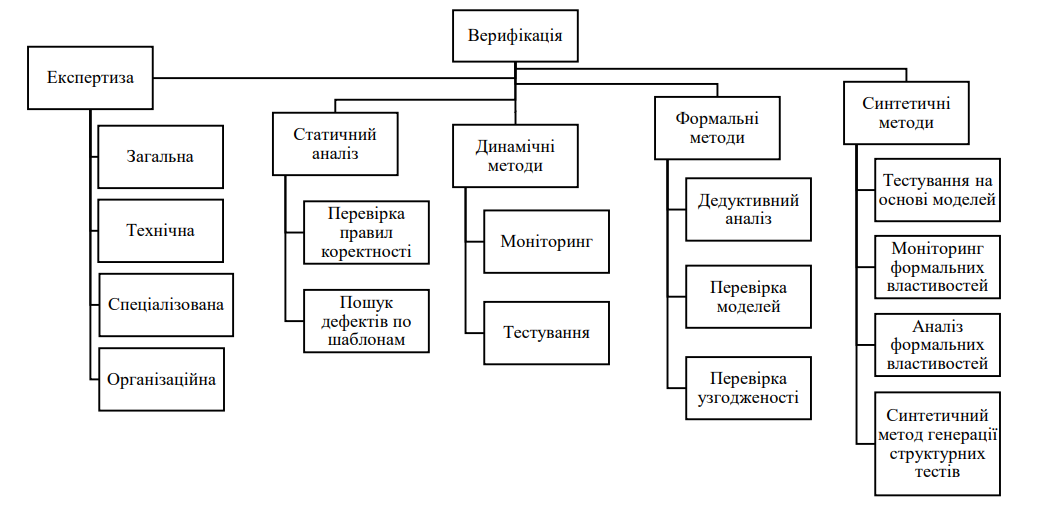


Рисунок 1.1 – Діаграма методів верифікації

## Огляд методів верифікації

Експертиза використовується для перевірки документації та коду програмного забезпечення на відповідність з нормами і стандартами оформленнями, які прийняті в країні, галузі та організації. Експертиза може бути загальною і спеціалізованою. Експертизою ПЗ також називають всі методи верифікації, в яких оцінка артефактів життєвого циклу ПЗ виконується людьми, що безпосередньо аналізують ці артефакти [1,4]. Перевагою даного методу є те, що при його використанні виявляються в середньому 50-90% помилок. Однак цей метод має і недоліки, оскільки для пошуку помилок, оцінки і аналізу властивостей ПЗ потрібні справжні експерти, а саме програмісти з досвідом роботи не менше 10 років.

Динамічні методи верифікації використовують результати реальної роботи програмної системи або її прототипів, щоб перевіряти відповідність цих результатів вимогам і проектним рішенням. Існує два основних види динамічних методів верифікації: моніторинг, в рамках якого йде тільки спостереження, запис і оцінка результатів роботи ПЗ при його звичайному використанні, і тестування, при якому ПЗ виконується в рамках заздалегідь підготовлених сценаріїв. Перевагою даного методу є висока точність виявлення помилок, а недоліками – необхідно мати набір вхідних даних та середовище виконання, а також високі вимоги до ресурсів[4,5].

Формальні методи верифікації. Їх істотною особливістю є можливість проведення пошуку помилок на математичній моделі, без звернення до фізичної реалізації, що в деяких випадках досить зручно і економічно. Для проведення аналізу формальних моделей застосовуються специфічні техніки, такі як дедуктивний аналіз, перевірка моделей, перевірка узгодженості. На жаль, для побудови таких моделей завжди необхідно виходити так само з коректності та адекватності моделі ПЗ [6]. Лише після правильної побудови цієї моделі можна автоматично проаналізувати деякі з її властивостей. Тож, в більшості випадків для ефективного аналізу від фахівців будуть потрібні глибокі знання математичної логіки і алгебри, а також деякого набору навичок роботи з цим апаратом.

Синтетичні методи верифікації є поєднанням підходів деяких вищезазначених методів верифікації, зокрема тестування на основі моделей, моніторинг формальних властивостей та синтетичний метод генерації структурних тестів. Переваги та недоліки синтетичних методів визначаються комбінацією методів верифікації, які входять до її складу.

## Статичний аналіз програмного забезпечення

Статичний аналіз – аналіз, що виконується без фактичного виконання програми. Методи статистичного аналізу можна розділити на два види: контроль того, що всі формалізовані правила коректності побудови цих артефактів виконані, та пошук типових помилок і дефектів в них на основі деяких шаблонів[1,4]. Часто інструменти статичного аналізу використовують обидва типи перевірок.

Статичний аналіз можна вважати найбільш широко застосовуваним методом верифікації. Перевірені на практиці правила коректності коду або шаблони типових помилок переносяться в середовища розробки.

Переваги статичного аналізу:

* автоматичний аналіз багатьох шляхів виконання одночасно. Виявлення помилок, що проявляються лише на одиничних шляхах виконання або на незвичайних вхідних даних;
* можливість аналізу на неповному наборі вхідних файлів. Відсутність накладних витрат під час виконання програми.

Недоліки даного методу:

* велика кількість помилкових спрацювань;
* необхідна ручна перевірка результатів роботи, що вимагає значних часових, людських та матеріальних ресурсів.

При статичній верифікації відбувається розбір тексту програми у її внутрішнє уявлення, в результаті якого будується граф потоку керування[1,4]. Генерація внутрішнього представлення програми відбувається в процесі синтаксичного аналізу та дозволяє зберегти її вихідну синтаксичну структуру. Вершини графа відповідають операторам, в тексті програми, а ребра асоціюються з передачею потоку управління.

Статичний аналіз забезпечує найбільш повне покриття коду у порівнянні з методами динамічної верифікації тільки в тому випадку, якщо верифікована програма не містить ділянок динамічно генерованого коду.

## Перевірка правил коректності

Методи статичного аналізу не залежать від використовуваного компілятора і середовища, що дозволяє знаходити приховані помилки, помилки не певної поведінки програми, легко виявляє помилки в тексті програми і помилки, викликані вставкою і копіювання різних частин коду. Але при цьому статичний аналіз досить слабо ефективний в діагностиці помилок, пов'язаних з витоком пам'яті, і має особливість генерувати велику кількість помилкових спрацьовувань, позначаючи всі підозрілі місця в тексті програми.

Проте сучасні методи володіють високою точністю і повнотою аналізу. Зокрема, майже у кожній сучасній IDE, не залежно від мови програмування є набір функціональних засобів, який дозволяє виконувати елементи статичного аналізу автоматично. Результатом такого аналізу є попередження та підказки, які допомагають розробнику писати більш зрозумілий і коректний код. Зокрема, використання статичних методів дозволяє знайти такі помилки, як [1,7]:

* невизначена поведінка програми, пов’язана з неініціалізованими змінними;
* змінні та методи, які ніколи не використовуються;
* звернення до NULL-вказівників;
* порушення правил і алгоритмів користування бібліотеками;
* сценарії, що призводять до недокументованої поведінки програми;
* переповнення буфера;
* помилки, що виникають в коді, що повторюється;
* помилки форматних рядків;
* помилки при використанні стандартних бібліотек тощо.

Верифікація методом статичного аналізу найбільш ефективна на етапі конструювання ПЗ, оскільки статичний аналіз застосовується до початкового тексту програми і не вимагає її виконання, що дозволяє істотно знизити вартість проекту і підвищити його надійність.

Інструменти автоматичної верифікації на основі статичного аналізу застосовуються досить широко, оскільки зручні і прості у використанні і не вимагають спеціальної підготовки програми на відміну від динамічного підходу.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Табунщик Г.В, Кудерметов Р.К., Каплієнко Т.І. / Т-12 Інженерія якості програмного забезпечення: навчальний посібник, 2-ге видання. – Запоріжжя: Дике Поле, 2016.. – 176 с.
2. Кулямин В.В. Методы верификации программного обеспечения. // Институт системного программирования РАН. 2008. – 117 с.
3. Інформаційні системи і технології на підприємствах: Тестування програм та систем: [Електронний ресурс] – режим доступу: https://pidruchniki.com/1628011847733/informatika/testuvannya\_program\_sistem
4. А.В. Лучкова / Аналіз методів верифікації програмного забезпечення: [Електронний ресурс] – режим доступу: https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/10955/328.pdf?sequence=3&isAllowed=y
5. Методы верификации и валидации сложных программных систем: [Електронний ресурс] – режим доступу: http://swsys.ru/index.php?page=article&id=3928
6. Формальна верифікація: [Електронний ресурс] – режим доступу: https://uk.wikipedia.org/wiki/Формальна\_верифікація
7. Статичний аналіз коду: [Електронний ресурс] – режим доступу: https://uk.wikipedia.org/wiki/Статичний\_аналіз\_коду