МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Інформаційна система тестування веб додатків

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям підготовки 6.050102 – "комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	Недзельський Д.О. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Трубніков В.М. (ініціали, прізвище)
Група:		КІ-14з

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 – "комп'ютерна інженерія"
(шифр і назва)
Спеціальність
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри КНІ
І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Трубнікову Владиславу Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна система тестування веб додатків

керівник проекту (роботи) Недзельський Дмитро Олександрович, к.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " " 201_ р. № ____
2. Термін подання студентом роботи 16.06.2018
3. Вихідні дані до роботи Теорія дерев рішень, найбільш популярні сайти

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз підходів до тестування програмного забезпечення, математична постановка задачі, методи розв'язання задачі та програмна реалізація, охорона праці

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

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст. викл. Критська Я.О.		

7. Дата видачі завдання 30.04.2018

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз завдання та робота з літературою	05.05.2018 - 13.05.2018	
2	Розробка інформаційної системи	14.05.2018 - 22.05.2018	
3	Тестування інформаційної системи	22.05.2018 - 02.06.2018	
4	Розробка розділу «Охорона праці»	02.06 .2018- 11.06.2018	
5	Оформлення пояснювальної записки та	11.06.2018 - 16.06.2018	
	електронних плакатів		

Здобувач вищої освіти _____ Трубніков В.М.
(підпис) (прізвище та ініціали)

Керівник _____ Недзельський Д.О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка дипломної роботи бакалавра: 69 с., 12 рис., 1 табл., 27 джерел, 1 додаток.

Робота присвячена розробці інформаційної системи тестування веб застосунків.

Був проведений аналіз проблеми недостатньої надійності продукції, а також проаналізовано різні підходи до тестування програмного забезпечення і вивчені різноманітні програми для проведення процесу тестування. У роботі розглянуті характеристики навантажувального тестування.

Була розроблена інформаційна система, на вхід якої подаються результати навантажувального тестування і за допомогою алгоритму побудови дерев рішень відбувається побудова нечітких правил, на підставі яких відбувається оцінка якості веб додатку.

Ключові слова: навантажувальне тестування, дерева рішень, web-додатка, критерій, атрибут.

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А,. м. Сєвєродонецьк, 93400.

ЗМІСТ

ВСТУП

Ми живемо у вік інформаційних технологій. Зараз у кожного з нас є техніка та вихід до Інтернету завдяки чому ми маємо можливість використовувати будь-які веб застосунки (ВЗ) у будь-який час, у будь-якому місці. То можуть бути ноутбуки або мобільні пристрої. Звичайно, є ті, хто розробляє дані застосунки, а також є ті, хто перевіряє наскільки якісно вони зроблені.

Щоб сказати, що даний ВЗ якісний, потрібно його протестувати. Для цього використовують цілий комплекс заходів, який допомагає виявити та виправити деякі помилки перед тим, як продукт потрапить до кінцевого користувача.

Сьогодні тестування проводиться не тільки після завершення фази конструювання. Воно проводиться на протязі всього процесу розробки. Це є дуже важливим етапом створення будь-якого програмного забезпечення, в тому числі для веб застосунків. Виявлення помилок не у кінці розробки, а на якомусь з етапів дозволяє виправити їх набагато швидше та с меншими переробками у всьому проекті.

«Якість продукції — це сукупність властивостей продукції, яку обумовлюють її придатність, задовольнити певні потреби відповідно до призначення»[1]. Основними параметрами якості вважаються: функціональна повнота, відповідність вимогам законодавства, безпека інформації, простота експлуатації, що не вимагає спеціальних знань в області інформаційних технологій, ергономічність користувальницького інтерфейсу, мінімізація витрат на експлуатацію, розвиток і модернізацію. Під надійністю ПЗ розуміють відсутність помилок, які б могли заважати нормальному функціонуванню підприємства.

Також важливо не тільки те, як ми тестуємо наш ВЗ, а й те, що ми

можемо сказати, отримавши результати.

Є багато програм та методів, які допомагають людині протестувати програмний продукт. В процесі тестування перевіряються певні характеристики розробляє мого продукту. Після проведення тестів людина отримує не тільки результат пройшов тест чи ні. Також людина може отримати певні числові дані. Наприклад, скільки людей можуть користуватися даним веб застосунком та він буде коректно працювати, скільки секунд проводилась загрузка ресурсів и т.д. Обробивши ці дані, спеціаліст може сказати наскільки якісно створений даний веб застосунок.

При розробці дипломного проекту слід враховувати небезпечні й шкідливі виробничі фактори, тому їх наявність знижує працездатність людини, а також може призвести як до травм, так і до професійних захворювань. Тому слід зменшити, а за можливості і усунути, вплив ОВПФ на людину.

1 АНАЛІЗ ПІДХОДІВ ДО ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Аналіз проблеми недостатньої надійності продукції

Дуже часто виникає така ситуація, коли користувач отримує якийсь програмний продукт та при використанні продукт працює некоректно та виникають якість незрозумілі для користувача помилки. Такі проблеми виникають через недостатню перевірку якості програмного продукту.

Саме для уникнення таких ситуацій потрібно дуже щільно тестувати програмний продукт. Програмісти, котрі розробляють даний продукт, не можуть протестувати його в повній мірі. Адже вони знають, що потрібно зробити або які дані потрібно ввести аби усе пройшло коректно, тому така перевірка не завжди є об’єктивною. Для цього наймаються тестувальники. Це люди, котрим віддають якесь програмне забезпечення для того, щоб вони його перевірили. Вони не знають, як саме зроблений той чи інший програмний продукт та по факту вони намагаються його зламати.

«Тестування програмного забезпечення — процес дослідження, випробування програмного продукту, що має дві різні цілі: продемонструвати розробникам і замовникам, що програма відповідає вимогам або виявити ситуації, в яких поведінка програми є неправильною, небажаною або не відповідає специфікації»[2].

Не існує єдиного унікального методу для тестування різних програмних засобів. Існує безліч методів та підходів до вирішення завдання тестування. Тому ефективне тестування програмних продуктів є творчим процесом, що не зводиться до слідування строгим і чітким процедурам. Проте є певні характеристики, котрим програмний продукт повинен відповідати:

- функціональність;
- надійність;

– зручність використання;

– ефективність;

– зручність супроводу;

– портативність.

Існує кілька ознак, за якими прийнято виробляти класифікацію видів тестування. Нижче наведені деякі з них.

1) За об'єктом тестування:

– функціональне тестування;

– тестування продуктивності (навантажувальне тестування, стрес-тестування, тестування стабільності);

– тестування зручності використання або юзабіліті-тестування;

– тестування інтерфейсу користувача;

– тестування безпеки;

– тестування локалізації;

– тестування сумісності.

2) За знанням системи:

– тестування чорної скриньки;

– тестування білої скриньки;

– тестування сірої скриньки.

3) За ступенем автоматизації:

– ручне тестування;

– автоматизоване тестування;

– напівавтоматизоване тестування.

4) За ступенем ізольованості компонентів:

– компонентне (модульне) тестування (component/unit testing);

– інтеграційне тестування (integration testing);

– системне тестування (system/end-to-end testing).

5) За часом проведення тестування:

– Альфа-тестування;

– Бета-тестування.

6) тощо.

Не останнім за важливістю є навантажувальне тестування. Термін тестування навантаження може бути використаний у різних значеннях в професійному середовищі тестування програмних засобів. У загальному випадку він означає практику моделювання очікуваного використання програми за допомогою емуляції роботи декількох користувачів одночасно. Тобто при навантажувальному тестуванні імітується робота багатьох користувачів, щоб побачити, як програмний продукт буде себе поводити у реальних обставинах. Подібне тестування найбільше підходить для багатокористувацьких систем, частіше - використовують клієнт-серверну архітектуру (наприклад, веб-серверів). Однак інші типи систем програмного забезпечення також можуть бути протестовані подібним способом. Наприклад, текстовий або графічний редактор можна змусити прочитати дуже великий документ.

Основною метою навантажувального тестування є те, щоб створити певне очікуване в системі навантаження (наприклад, за допомогою віртуальних користувачів) і спостерігати за показниками продуктивності системи.

В ідеальному випадку в якості критеріїв успішності навантажувального тестування виступають вимоги до продуктивності системи. Однак часто буває так, що такі вимоги не були чітко сформульовані. У цьому випадку перше навантажувальне тестування буде пробним і буде ґрунтуватися на розумних припущеннях про очікуване навантаження.

Отже, навантажувальне тестування — дуже важливий етап у розробці будь-якого програмного продукту. У тому числі і для веб застосунків, адже розробники повинні усвідомлювати скільки користувачів можуть користуватися їх застосунком та він при цьому буде коректно працювати. І дуже важливо розуміти при скількох користувачах застосунок може «впасти», щоб спробувати якось цьому запобігти.

1.2 Огляд програм для навантажувального тестування

На сьогодні існує дуже багато програм для навантажувального тестування. Звичайно, усі ці програми неможливо використовувати одночасно. Тож нижче наведені лише деякі з них.

Apache JMeter програма є Java-додатком з відкритим кодом. Вона призначена для навантажувального тестування веб-додатків та їх окремих компонентів, а також для тестування FTP-серверів, баз даних (з використанням JDBC) та мережі. Під компонентами веб-додатків слід розуміти скрипти, сервлети, Java об'єкти тощо.

Apache JMeter розробляється Apache Software Foundation та стабільно випускається з 2014 року. Написана дана програма на мові програмування Java, тому є кросплатформеною. Це означає, що вона працює в різних *nix-системах, в Windows від 98 і деяких інших ОС.

У JMeter передбачені механізми авторизації віртуальних користувачів, підтримуються користувальницькі сеанси, шаблони, хешування і подальший offline аналіз результатів тесту, функції дозволяють сформувати наступний запит, ґрунтуючись на відповіді сервера на попередній. При виконанні тестування користувач має можливість використовувати проксі-сервер, який вбудований до JMeter та призначений для запису сесій, а може використовувати будь-який зовнішній.

Перед тим, як почати тестування потрібно скласти тестовий план, що описує серію завдань, які необхідно виконати JMeter. Він повинен містити одну або кілька груп потоків (Thread Groups) і інші елементи:

- логічні контролери (Logic controllers);
- типові контролери (Sample generating controllers);
- слухачі (Listeners);
- таймери (Timers);
- відповідності (Assertions);

– конфігураційні елементи (Configuration elements).

Насамперед потрібно створити групу потоків (Edit - Add - Thread Group). У її налаштуваннях ми маємо вказати назву, кількість потоків, що запускаються, тобто віртуальних користувачів (Number of threads), час затримки між запуском потоків (Ramp-Up Period) та кількість циклів виконання завдання (Loop Count). Для тестування навантаження або перевірки працездатності сервера досить вибрати HTTP Request (Add-Sampler - HTTP Request). Тут вказуємо назву, IP-адресу та порт веб-сервера, протокол, метод передачі даних (GET, POST), параметри переадресації, передачу файлів на сервер. Налаштовуємо і тиснемо на Run. Після цього почнеться процес тестування нашого веб застосунку.

JMeter дуже потужний інструмент тестування, проте він має одну велику ваду: неосвідченному користувачеві дуже складно розібратися у всіх тонкощах даної програми та правильно її настроїти.

Як виглядає JMeter показано на рис. 1.1.

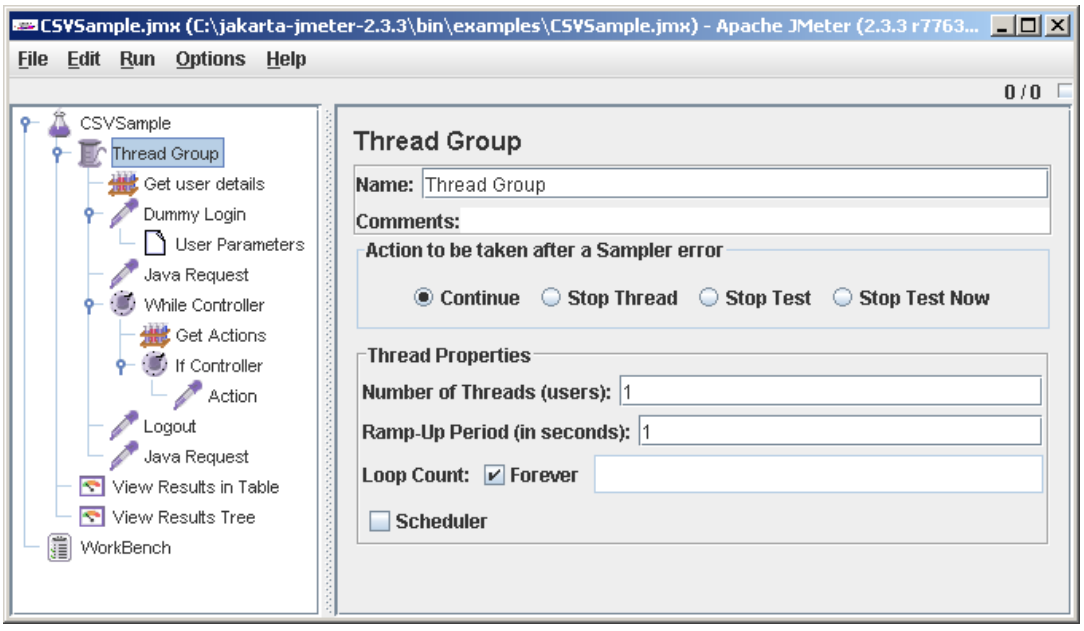


Рисунок 1.1 – Графічний інтерфейс Apache JMeter

HP LoadRunner — утиліта для автоматизованого навантажувального тестування. Програма може виконувати як тестування різних додатків, так і

тестування сайтів різного рівня складності. При підключенні віртуальних користувачів виконуються різні скрипти, за різними сценаріями. Програма має відповідні набори інструментів для проведення тестування. Так само до складу HP LoadRunner входить набір інструментів для роботи по різних протоколах з додатком (віддалено, через проксі-сервер і т.п.).

HP LoadRunner складається з наступних додатків:

– virtual user generator (vugen) - служить для розробки навантажувальних скриптів;

– load generator - служить для генерації навантаження (генерації віртуальних користувачів);

– controller - служить для розробки і запуску сценаріїв навантаження;

– analysis - служить для аналізу результатів тестування навантаження.

Як виглядає HP LoadRunner показано на рис. 1.2.

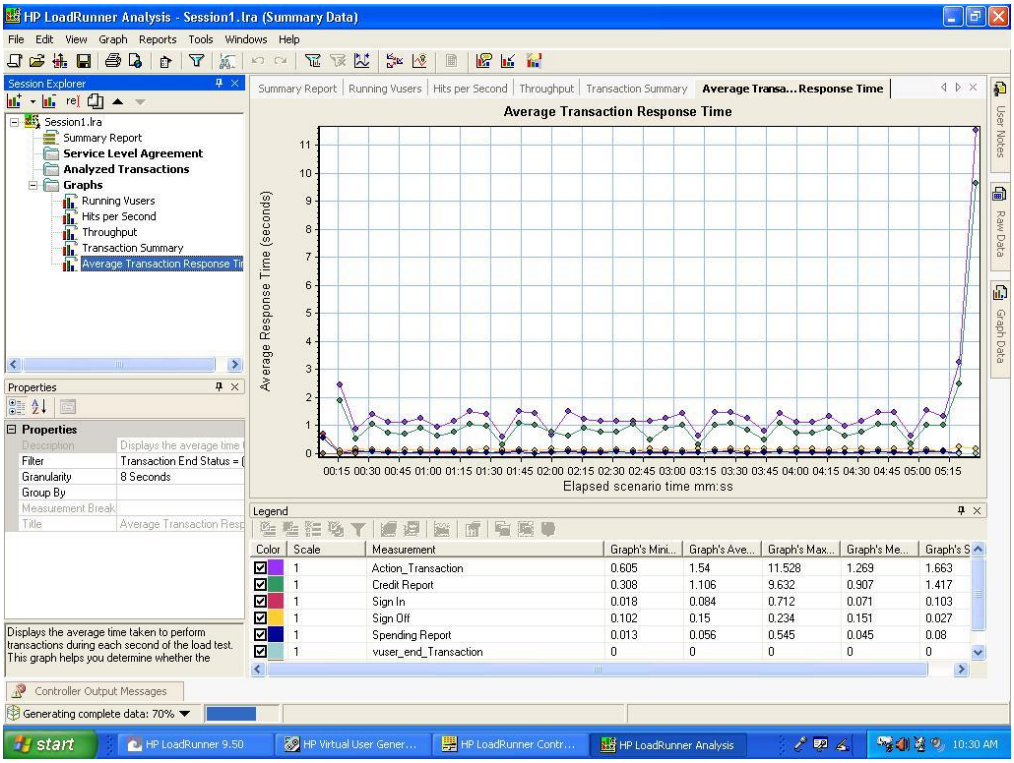


Рисунок 1.2 – Вигляд програми HP LoadRunner

IBM Rational Performance Tester дає змогу, завдяки правильно сформульованим тестам, використовувати поточні навантаження для оцінки

навантажень, можливих у майбутньому. Це означає, що якщо якийсь веб додаток може обслуговувати 500 користувачів, то завдяки даній програмі ми можемо змоделювати навантаження в 1000 користувачів. Завдяки цьому ми можемо більш точно визначити вимоги до сервера.

Ми маємо можливість включити такі елементи до нашого плану тестування: час на роздуми користувача, час затримки, цикл, перемикач рандомізації тощо. IBM Rational Performance Tester розробляється фирмою Rational Softwareта працює на комп'ютерах, що працюють під управлінням Windows NT/2000/XP та Linux. Як виглядає IBM Rational Performance Tester показано на рис. 1.3.

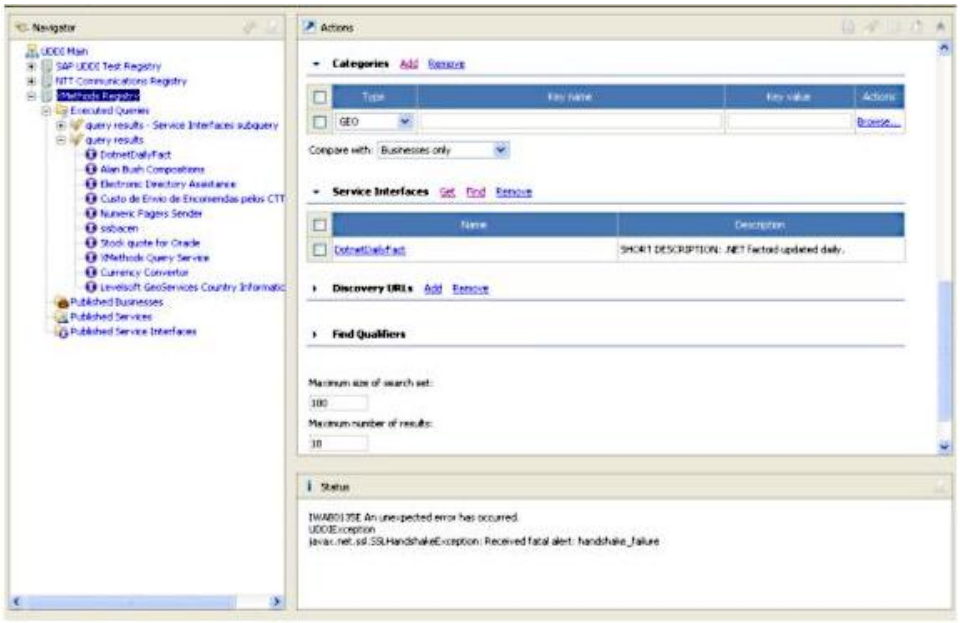


Рисунок 1.3 – Графічний інтерфейс IBM Rational Performance Tester

Програмний продукт WAPTдозволяє випробувати стійкість веб-додатку до реальних навантажень. WAPT розробляється новосибірської компанією SoftLogica LLC. Працює під управлінням Windows. Завдяки WAPT ми маємо можливість створити безліч віртуальних користувачів та надати кожному індивідуальні параметри. Даний інструмент дозволяє встановлювати затримки між запитами та динамічно генерувати деякі випробувальні параметри. Завдяки цьому ми маємо можливість імітувати

діяльність реальних користувачів.

У запит ми можемо підставити різні варіанти HTTP-заголовка, а в налаштуваннях можна вказати кодування сторінок. Ми маємо можливість вказати параметри User-Agent, X-Forwarded-For, IP в налаштуваннях сценарію. Значення параметрів запиту можуть бути розраховані кількома способами. Наприклад, визначені відповіддю сервера на попередній запит, використовуючи змінні і функції.

У даному програмному продукті підтримується робота по захищеному протоколу HTTPS (і всі типи проксі-серверів). Сценарії, які ми створимо, будуть зберігаються у файлі XML-формату. В списку тестів присутні стандартні Performance і Stress, а також кілька інших тестів. Все це дозволяє визначити максимальну кількість користувачів і тестувати сервер під навантаженням протягом довгого періоду.

Зовнішній вигляд WAPT наведено на рис. 1.4.

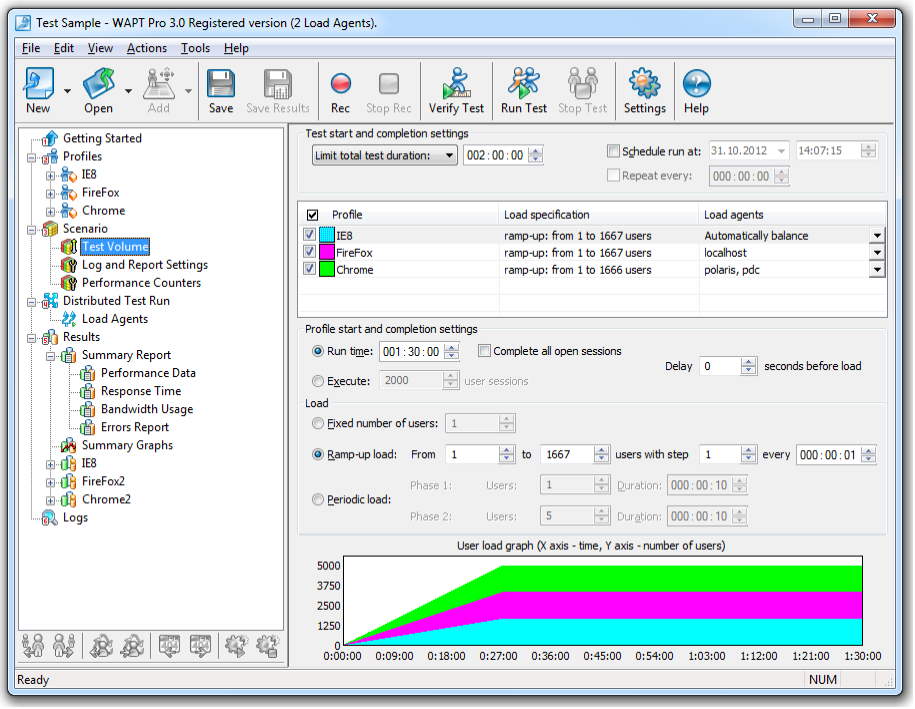


Рисунок 1.4 – Інтерфейс WAPT

Opensource інструмент Pylot має великий набір фукнціонального переваг, до того ж відкритий вихідний код дозволяє доопрацювати його під

конкретні потреби.

Даний інструмент написан з використання мови програмування Python та є кроссплатформеним.

Основні можливості Pylot:

– підтримка протоколів HTTP і HTTPS (SSL);

– генерація багатопотокової навантаження;

– автоматична обробка cookies;

– перевірка відповідей сервера за допомогою регулярних виразів;

– відображення статистики в реальному часі;

– генерація докладного звіту про результати тестування (HTML + графіки);

– робота в консольному і графічному режимах;

– можливість віддаленої роботи за допомогою XML-RPC.

Графічний інтерфейс програми наведено на рис. 1.5.

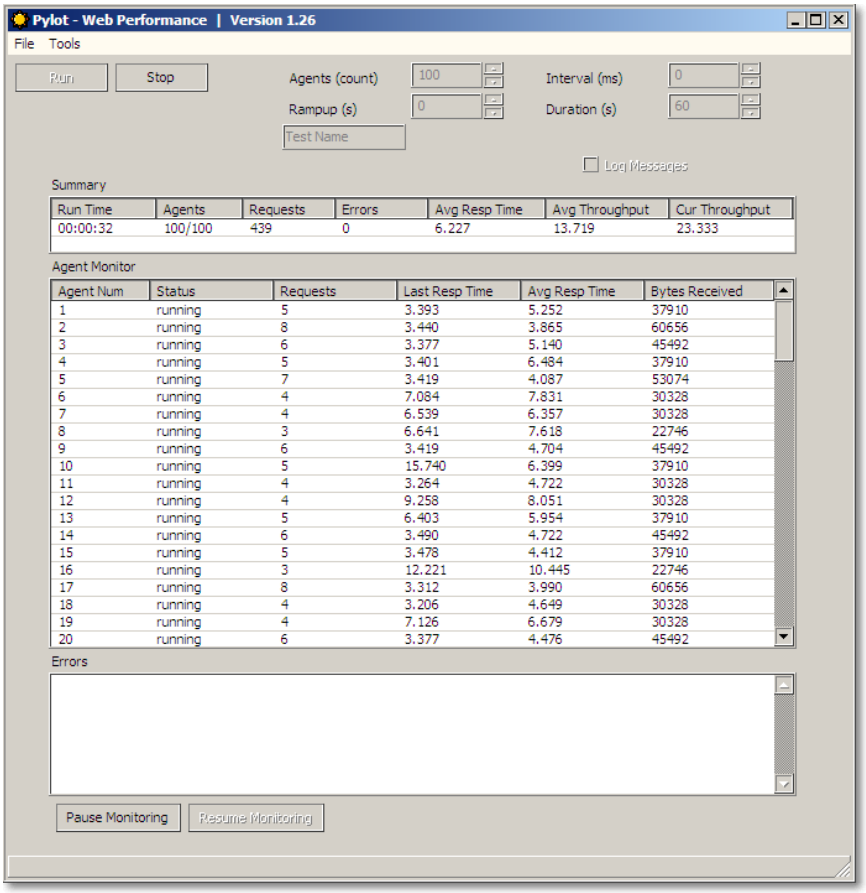


Рисунок 1.5 – Графічний інтерфейс Pylot

Програмний продукт Go-meter, що є доповненням JMeter. JMeter не міг створити дуже великий трафік, тому вирішуючи дану проблему було створено нову програму. Даний інструмент написан с використанням мови програмування Go. Усі тести зберігаються у JSON форматі. Після запуску програми читається профіль і запускається тестування. Під час тестування до консолі виводиться зведена таблиця. У цій таблиці формується світ про тестування: час відповіді, кількість запитів в секунду і процентне відношення помилок, попереджень і вдалих запитів. Тестування за допомогою даного інструменту відбувається шляхом створення структури для кожного елемента, відкриття та зчитування файлу.

Консольний мультіпотоковий Siege додаток для навантажувального тестування веб-серверів. Даний інструмент є мультиплатформенним і доступним під усіма сучасними ОС. Дана програма дає можливість перевірити ресурсомісткість свого коду в умовах, максимально наближених до реальних.

Siege може імітувати звернення до сайту відразу декількох користувачів. Це дозволяє тримати сервер як би «під облогою» довгий час. Кількість запитів, зроблених при «облозі», розраховується із загальної кількості користувачів і кількості їх звернень до сервера. Результат тестування має: час витрачений на перевірку, загальна кількість переданої інформації (включаючи заголовки), середній час відповіді сервера, його пропускну здатність і число запитів на які прийшла відповідь з кодом 200. Ці дані формуються і видаються при кожній перевірці.

Siege має 3 основних моделі роботи: режим регресійного тестування, режим імітації Інтернету і режим грубої сили. Програма зчитує порцію посилань з конфігураційного файлу і звертається до них по черзі або випадково. Також користувач може вказати один єдиний адресу до якого будуть проводитися всі звернення.

Існує ще дуже багато програм для навантажувального тестування, проте основні з них та найбільш популярні були перераховані та

охарактеризовані.

Кожна з описаних програм має свої переваги та недоліки. Звичайно, що неможливо вивчити їх усі. Тому компанії обирають ту програму, яка задовольняє їх потреби та дозволяє вирішувати усі поставлені перед нею задачі. На підставі цього можна сказати, що навіть однієї програми достатньо для гарної перевірки програмного забезпечення.

1.3 Постанова задачі

Метою даної роботи є автоматизація обробки даних, отриманих у результаті навантажувального тестування веб додатків. Для реалізації нам необхідно:

1) провести аналіз підходів до тестування;
2) розробити вимоги до продуктивності веб додатків;
3) розробити програму, яка буде проводити аналіз даних, які будуть отримані у результаті проведення навантажувального тестування веб додатків;
4) розробити правила нечіткої логіки, які видавали б результат на скільки ефективно сайт витримує навантажувальне тестування, на основі параметрів, отриманих з програми.

2 МАТЕМАТИЧНА ПОСТАНОВА ЗАДАЧІ

2.1 Характеристики навантажувального тестування

«Навантажувальне тестування — підвид тестування продуктивності, збір показників і визначення продуктивності та часу відгуку програмно-технічної системи або пристрою у відповідь на зовнішній запит з метою встановлення відповідності вимогам, що пред'являються до даної системи»[3]. Як ми бачимо із визначення навантажувальне тестування дозволяє оцінити багато різноманітних характеристик. Деякі з них опишемо нижче:

1) характеристика помилок.

По-перше, це ті помилки, що виникли протягом тесту, та їх опис. Ініціалізація помилки відбувається завдяки коду помилки. По-друге, це кількість помилок, що виникли протягом тесту. Відзначається кількість помилок, що виникли на протязі тесту за секунду, та загальна кількість помилок протягом усього процесу тестування.

2) Характеристики сценарію.

При виконанні певного сценарію створюються віртуальні користувачі, які імітують діяльність реальних користувачів. Характеристики сценарію дозволяють виявити поведінку таких користувачів протягом усього сценарію. Відстежують відносний час (тимчасова шкала, в якій за нуль прийнято час початку тесту) та кількість віртуальних користувачів (загальне або в певний час тесту).

3) Характеристики безпеки;

Дані характеристики показують інформацію про атаки на сервер протягом виконання сесії. Дана інформація представляється у вигляді кількості пакетів.

4) Характеристики web-ресурсів.

Дані характеристики несуть інформацію про дії користувачів в секунду (кількість HTTP-запитів, які були надіслані віртуальними користувачами на web-сервер), HTTP-відповіді, пропускна здатність (скільки даних може пропустити через себе сервер в секунду) та з'єднання (кількість відкритих TCP / IP з'єднань в кожній точці часу сценарію), а також SSL-з'єднання (відкриваються браузером після того, як TCP/IP- з'єднання відкрилося на сервер безпеки).

5) Характеристики декомпозиції сторінок.

Дані характеристики містять час завантаження компонент сторінок, час відгуку кожної Web-сторінки і її компонент протягом тесту, розмір завантажених компонент сторінок та декомпозицію часу завантаження сторінок і їх компонентів (час на визначення доменного імені, час на з'єднання з сервером, час отримання, клієнтське час, час помилки).

6) Характеристики транзакцій.

Дані характеристики мають інформацію про загальна кількість, транзакції в секунду та час на виконання транзакції.

7) Ресурси Web-серверів.

Дані характеристики містять інформацію про використання ресурсів Web-серверів Apache, Microsoft IIS, iPlanet / Netscape, and iPlanet (SNMP), а також серверів баз даних, серверів ERP / CRM і серверних додатків.

8) Мережева конфігурація.

Дані характеристики використовуються для визначення, коли мережа викликає затримку в сценарії, проблемний мережевий сегмент. Це первинний фактор в продуктивності додатків і Web-систем.

9) Характеристики системних ресурсів:

– ресурси Windows (відсоток загального процесорного часу, процесорний час,% - відсоток часу, який витрачають всі процесори системи на виконання певного потоку, операції з файлами даних в секунду, потоки - кількість потоків на комп'ютері під час збирання даних, приватні біти -

поточне число приватних бітів, які визначені процесом, недоступні для інших процесів);

– ресурси UNIX (середнє завантаження, інтенсивність зіткнень (collision rate), частота контекстного комутатора (context switch rate), використання ЦП, частота помилкових вхідних / вихідних пакетів, частота вхідних / вихідних пакетів, частота переривань, частота сторінкової підкачки файлів, швидкість підкачки / розвантаження (SWAP), використання ЦП в системному і користувальницькому режимах);

– серверні ресурси (ресурси (завантаження ЦП, пам'ять, дисковий простір, служби), що використовуються віддаленим сервером, вимірювані протягом сценарію);

– SNMP-ресурси (статистика для машин, що працюють по SNMP простий протокол мережевого управління).

Це основні характеристики навантажувального тестування.

2.2 Постановка задачі автоматизації аналізу результатів навантажувального тестування web – систем

Виходячи з проведеного системного аналізу проблеми тестування програмного забезпечення, було визначено, що найбільш пріоритетним (ефективним) серед різних видів тестування є тестування навантаження. В рамках даної дипломної роботи сформульована задача оцінки якості програмного забезпечення на основі показників, отриманих в результаті проведення навантажувального тестування.

Критерієм ефективності функціонування інформаційної системи (сайту) є наступне:

$$I(X,Z,U,V) \to \max_{U \in G}, \tag{2.1}$$

де I – інтегральна оцінка показників ефективності.

І із формули (2.1) вираховується із формули:

$$I = \sum_{i=1}^{n} \lambda_i p_i(x_i, U), \tag{2.2}$$

де λ_i – показник значущості і-го показника ефективності,

p_i – показник ефективності для і-го типу тесту (сценарію).

p_i витікає із формули:

$$p_i(x_i, U) = \varphi_i(S(U)), \tag{2.3}$$

де $S = (X, Z)$ – вектор стану системи,

$Z = (z_1, z_2, ..., z_K)$ – технічні характеристики системи,

$X = (x_1, x_2, ..., x_M)$ – характеристики результатів сценаріїв навантажувального тестування.

Характеристики X рахуються за формулою:

$$x_i = \psi_i(Z(U)), \tag{2.4}$$

де $U = (u_1, u_2, ..., u_L)$ – можливі управляючі дії (рекомендації) щодо оптимізації якості інформаційної системи,

$V = (v_1, v_2, ..., v_N)$ – вимоги до продуктивності інформаційної системи.

2.3 Опис типів навантажувального тестування

Основними принципами навантажувального тестування є: оцінка продуктивності та працездатності.

Навантажувальне тестування, це тестування, у ході якого перевіряється продуктивність і працездатність системи при різному навантаженні й при різних системних ресурсах. Як мета тестування висувається перевірка на відповідність характеристик системи значенням, необхідним специфікацією, загальноприйнятим нормам і здоровому глузду. Для навантаження системи при тестуванні використають різні вхідні дані й варіанти тестового оточення. Властиво навантажувальне тестування може бути розділене на кілька типів.

1) Тестування продуктивності.

Тестування продуктивності виконується з метою визначити тимчасові характеристики системи при заданому робочому навантаженні. Тестування продуктивності має кілька цілей. Воно показує, що система відповідає критеріям продуктивності; указує на частини, що мають найменшу продуктивність; може продемонструвати переваги даної системи над аналогічною. При розробці систем, критичних до продуктивності, даний вид тестування стартує на самому початку розробки й виробляється аж до розгортання системи.

Тобто при даному типі тестування ми повинні перевірити, як наш веб додаток буде себе поводити при заданій кількості користувачів. Звичайно, в нас є обмеження на час, який потрібен для завантаження сторінки. При проведені тестування час на завантаження сторінки при заданій кількості користувачів не повинен перебільшувати допустимий час.

Якщо ми побудуємо графік результатів, то в нас будуть два графіки: графік, коли час на завантаження сторінки є допустимим, і графік, який буде показувати фактичний час завантаження сторінки при заданій кількості користувачів. Ці два графіки можуть бути майже однаковими, а можуть бути зовсім різними. Ми маємо ураховувати тангенс кута між нашим графіком та

віссю Ох. Так як ми знаємо допустимий час на завантаження сторінки, ми можемо отримати також допустиме значення тангенсу кута. Тож тангенс кута графіку, який ми отримаємо у результаті, не повинен перебільшувати допустиме значення.

Таким чином, у нас є два параметри, котрі ми повинні перевірити при тестуванні. Якщо результати тестування не задовольняють хоча б одному з ним, ми можемо зробити висновок, що наш веб застосунок спроектовано не досить якісно.

2) Тестування одночасного доступу до даних (Contention Tests).

Цей вид тестування націлений на знаходження вузьких місць у продуктивності (блокування, витоку пам'яті, «пробуксовки» у віртуальній пам'яті), викликаних певною кількістю віртуальних користувачів, що звертаються до тим самим ресурсів.

Тест визначає, який максимальний сплеск у часі виконання транзакцій може витримати додаток без помилок. Виміряються мінімальний, середній і максимальний час на кожну дію.

При цьому типі тестування ми повинні перевірити ті самі критерії, що й при тестуванні продуктивності. Проте перевіряються лише сторінки,де є підключення до бази даних.

3) Тестування навантаження.

При навантажувальному тестуванні виробляється перевірка працездатності системи при зміні навантаження на систему: кількості користувачів, транзакцій і т.п. Дозволяє визначити границі, при яких характеристики системи залишаються на прийнятному рівні, що дозволяє продовжувати роботу з нею.

Тобто ми збільшуємо кількість користувачів та дивимось скільки часу потрібно для завантаження сторінки нашого веб додатку. На підставі отриманих даних ми можемо побудувати графік залежності часу завантаження сторінки від кількості користувачів, котрі зараз знаходяться на

цій сторінці. Завдяки цьому ми можемо на графіку знайти допустимий час та побачити скільки користувачів можуть знаходитись на сторінці одночасно.

4) Тестування стійкості (Endurance "Soak" "Longevity" Tests).

Цей вид навантажувального тестування перевіряє, чи може система підтримувати постійну кількість віртуальних користувачів, що виконують транзакції, максимально використовуючи пропускну здатність системи.

Даний тест дозволяє виявити «накопичувальні» помилки в коді, а також трапляються час від часу події й аномалії. Також перевіряються події, які спрацьовують за розкладом.

Однією із цілей тестування є визначення схильності системи до деградації (наприклад, наявність витоків пам'яті). Для цього система залишається під навантаженням на тривалий час, а потім перевіряється, чи змінилися її тимчасові характеристики, розміри, займані в пам'яті.

Для цього типу тестування справедливі критерії із пунктів 1 та 2.

5) Стресове тестування.

Ціль стресового тестування визначити, при яких навантаженнях системи відбувається її відмова, причини цієї відмови, поводження системи в цьому випадку, коректність збереження й обробки даних. Існує кілька підходів до реалізації стресового тестування - це збільшення навантаження на систему аж до відмови й урізування ресурсів системи (оперативної пам'яті, дискового простору, процесорних ресурсів). Можна комбінувати дані підходи для якнайшвидшого досягнення критичної крапки.

Стресове тестування дозволяє визначити мінімальні величини ресурсів, необхідних для роботи додатка, оцінити граничні можливості системи і виявити фактори, що обмежують ці можливості (тобто знайти «вузькі місця» системи). Крім того, метою стресового тестування може бути виявлення помилок у серверному додатку, а також тестування здатності системи до збереження цілісності даних в аварійних ситуаціях і до відновлення після таких ситуацій.

При цьому типі тестування ми перевіряємо, як буде поводити себе система при кількості користувачів більшій, ніж допустима. При аналізі побудованих графіків ми маємо ураховувати, що відправна точка β_0 повинна бути меншою за допустимий час, а тангенс кута графіку, який ми отримаємо у результаті, не повинен перебільшувати допустиме значення. Крім того, кореляція між графіками r повина бути меншою за припустиме значення.

6) Об'ємне тестування.

Об'ємне тестування, або тестування обсягу - вид тестування, при якому на обробку в систему подаються більші обсяги даних і вивчається поводження системи при цьому. Наприклад, обробка більших файлів у текстовому, більших зображень у графічному редакторі.

7) Тестування масштабованості.

Цей вид навантажувального тестування має на увазі повторення попередніх видів тестів на різних конфігураціях серверного/мережного встаткування з метою визначення найефективнішого за ціною варіанта, що буде задовольняти поставленим вимогам продуктивності.

2.4 Методи розрахунку

2.4.1 Метод оцінки взаємозв’язків факторів

Нехай X - незалежна змінна (фактор), Y - залежна змінна (відгук). Якщо до цього часу ми розглядали вибірку з однією змінною, то тепер ми маємо вибірку парних спостережень (x1, y1), (x2, y2), ..., (xn, yn).

Вибірка може бути отримана одним із двох способів: або це вимірювання параметрів одного об'єкта (технологічного процесу, технологічної системи - літака) в моменти часу t1, t2, ..., tn, або ми обстежили n об'єктів і для кожного з них виміряли деякі параметри (x, y).

Припустимо, ми припускаємо що є лінійна залежність між Y і X. Наприклад, залежність ваги від зростання (було обстежено n осіб). Незалежно від способу отримання вибірки існує 2 попередніх кроку для визначення існування і ступеня лінійної залежності між X і Y.

1-й крок. Це графічне зображення точок (x1, y1), ..., (xn, yn) на площині XY.

Такий графік називається діаграмою розсіювання.

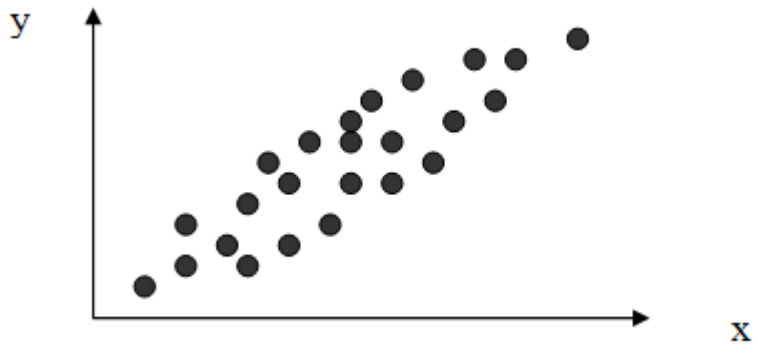


Рисунок 2.1 – Діаграма розсіювання

Візуально аналізуючи діаграму ми можемо емпірично вирішити, чи припустимо припущення про лінійну залежність між X і Y.

2-й крок. Це обчислення вибіркового коефіцієнта кореляції.

Коефіцієнт кореляції між X і Y→rXY:

$$r_{XY} = \frac{S_{XY}}{S_x \cdot S_Y}, \qquad (2.5)$$

де S_{XY} ковариация між X і Y,

$S_X = \sqrt{S_y^2}$ - корінь з дисперсії,

$S_y = \sqrt{S_y^2}$ - корінь з дисперсії.

Згадаємо, що дисперсія:

$$S_x^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 . \quad (2.6)$$

Тоді:

$$S_y^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2 . \quad (2.7)$$

Тоді S_{xy}^2:

$$S_{xy}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}); -1 \le r_{xy} = r_{yx} \le 1. \quad (2.8)$$

Чим ближче |rxy| к 1, тим більша ступінь лінійної залежності.
Чим ближче rxy к 0, тим вона менша. Тож:

$$r_{xy}^2 = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})[\sum_{i=1}^{n}(x_i - \bar{x})^2 \cdot \sum_{i=1}^{n}(y_i - \bar{y})^2]. \quad (2.9)$$

2.4.2 Метод визначення граничних параметрів системи

Ми можемо побудувати графік, що відображає динаміку вихідного процесу.

За цим графіком будується ковзне середнє:

$$P'_i = \frac{1}{m}\sum_{k=1}^{m} P_k \ , \tag{2.10}$$

де m – розмір вікна усереднення;

P_k – вихідні значення показника;

P'_i – значення показника після усереднення.

Для згладженого часового ряду значень показника будується лінійна регресія:

$$Y = \beta_0 + \beta_1 X + e, \tag{2.11}$$

де Y - залежна змінна (відомий вектор);

X - незалежна змінна (відомий вектор);

β_0, β_1 - параметри моделі, які необхідно оцінити;

e - вектор помилок.

У загальному випадку незалежних змінних може бути кінцеве число.

Оцінка параметрів моделі проводиться згідно з методом МНК, щоб при підстановці оцінок в рівняння лінійної моделі величина помилок була мінімальною.

Суть методу МНК детально представлена далі.

Розглянемо рівняння:

$$y_i = \beta_0 + \beta_1 x_i + e_i, \ i = \overline{1,n}. \tag{2.12}$$

Найкращими оцінками коефіцієнтів β_0 та β_1 будуть оцінки $\hat{\beta}_0$ та $\hat{\beta}_1$, що доставляють мінімум сумі квадратів залишків.

Розрахуємо їх:

$$S(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_1 x_i - \beta_0)^2 \to \min . \quad (2.13)$$

Якщо $S(\beta_0, \beta_1)$ має похідні по β_1 та β_0, то необхідною умовою мінімуму є рівняння (2.14).

Аналізуючи коефіцієнт, можна судити про тенденції часу завантаження.

Якщо $\hat{\beta}_1 \approx 0$, то можна стверджувати, що на заданому інтервалі критеріїв продуктивності час завантаження не змінюється зі збільшенням навантаження, тобто система успішно витримує навантаження в термінах досліджуваних величин.

У випадку, коли $\hat{\beta}_1 > 0$ можна судити про падіння продуктивності системи, що проявляється у вигляді збільшення часу завантаження сторінок при збільшенні навантаження.

Порахуємо умову мінімуму для рівняння (2.13):

$$\frac{\partial S}{\partial \beta_1} = -2\sum_{i=1}^{n} (y_i - x_i \hat{\beta}_1 - \hat{\beta}_0) x_i = 0 . \quad (2.14)$$

$$\frac{\partial S}{\partial \beta_0} = -2\sum_{i=1}^{n} (y_i - x_i \hat{\beta}_1 - \hat{\beta}_0) = 0 . \quad (2.15)$$

Таким чином, отримуємо систему рівнянь:

$$\begin{cases} \sum_{i=1}^{n} (y_i - x_i \hat{\beta}_1 - \hat{\beta}_0) x_i = 0, \\ \sum_{i=1}^{n} (y_i - x_i \hat{\beta}_1 - \hat{\beta}_0) = 0. \end{cases} \quad (2.16)$$

Вирішуючи систему рівнянь, отримуємо оцінки параметрів лінійної моделі:

$$\begin{cases} \hat{\beta}_0 = \bar{y} - \beta_1 \bar{x} \\ \hat{\beta}_1 = \dfrac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2}, \end{cases} \tag{2.17}$$

де $\bar{x}$ и $\bar{y}$ - середні значення векторів X і Y відповідно.

Досить важливим є питання встановлення моменту, в який графік починає зростати - значення кількості, відповідне цьому моменту характеризує граничне навантаження системи, тобто граничне число користувачів.

Для визначення цієї точки вважаємо точку, в якій похідна усередненого ряду буде міняти знак. Похідну при цьому вважаємо чисельно, і відповідно з деяким порогом вважаємо зміна знака похідної.

3 МЕТОДИ РОЗВ'ЯЗАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Алгоритм дерева рішень

3.1.1 Алгоритм

Нехай нам задано множину прикладів T, де кожен елемент цієї множини описується m атрибутами. Кількість прикладів в безлічі T будемо називати потужністю цієї множини і позначимо $|T|$. Нехай мітка класу приймає наступні значення C1, C2 ... Ck.

Задача полягає у побудові ієрархічної класифікаційної моделі у вигляді дерева з множині прикладів T. Процес побудови дерева відбувається зверху вниз. Спочатку створюється корінь дерева, потім нащадки кореня тощо.

На першому кроці мається порожнє дерево (тільки корінь) і вихідна множина T (асоційована з коренем). Потрібно розбити початкову множину на підмножини. Вибирається один з атрибутів в якості перевірки. Тоді в результаті розбиття виходять n (по числу значень атрибута) підмножин і, відповідно, створюються n нащадків кореня, кожному з яких відповідає своя підмножина, отримана при розбитті множини T. Потім ця процедура рекурсивно застосовується до всіх підмножини (нащадкам кореня) і т.д.

Розглянемо докладніше критерій вибору атрибута, за яким має піти розгалуження. Мається m (по числу атрибутів) можливих варіантів, з яких треба вибрати той, що найбільш підходить. Деякі алгоритми виключають повторне використання атрибуту при побудові дерева, але у даному випадку таких обмежень не накладається, тобто будь-який з атрибутів можна використовувати необмежену кількість разів при побудові дерева.

Нехай маємо перевірку X (в якості перевірки може бути обраний будь-який атрибут), яка приймає n значень A1, A2 ... An. Тоді розбиття T з перевірки X дасть підмножини T1, T2 ... Tn, при X рівному відповідно A1, A2 ... An. Єдина доступна тут інформація - те, яким чином класи розподілені

у множині T і його підмножинах, що одержані при розбитті по X. Саме це використовується при визначенні критерію.

Нехай freq(Cj,S) - кількість прикладів з деякої множини S, які стосуються одного й того ж класу Cj. Тоді ймовірність того, що випадково обраний приклад з безлічі S буде належати до класу Cj:

$$P = \frac{freq(C_j,S)}{|S|} . \quad (3.1)$$

Згідно теорії інформації, кількість міститься у повідомленні інформації, залежить від її ймовірності:

$$log_2\left(\frac{1}{P}\right) . \quad (3.2)$$

Оскільки ми використовуємо логарифм з двійковим підставою, то вираз (3.1) дає кількісну оцінку в бітах.

$$Info(T) = -\sum_{j=1}^{k} \frac{freq(C_j,T)}{|T|} * log_2\left(\frac{freq(C_j,T)}{|T|}\right) . \quad (3.3)$$

Вираз дає оцінку середньої кількості інформації, необхідної для визначення класу прикладу з множини T. У термінології теорії інформації вираз (3.3) називається ентропією множини T.

Ту ж оцінку, але тільки вже після розбиття множини T по X, дає такий вираз:

$$Info_X(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} * Info(T_i) . \quad (3.4)$$

Тоді критерієм для вибору атрибута буде наступна формула:

$$Gain(X) = Info(T) - Info_X(T) \tag{3.5}$$

Критерій (3.5) вважається для всіх атрибутів. Вибирається атрибут, який максимізує даний вираз. Цей атрибут буде перевіркою в поточному вузлі дерева, а потім по цьому атрибуту проводитиметься подальша побудова дерева. Тобто у вузлі буде перевірятися значення по цьому атрибуту і подальший рух по дереву буде проводитися в залежності від отриманої відповіді.

Такі ж міркування можна застосувати до отриманих підмножин T1, T2 ... Tn і продовжити рекурсивно процес побудови дерева, до тих пір, поки у вузлі не опиняться приклади з одного класу.

Одне важливе зауваження: якщо в процесі роботи алгоритму отриманий вузол, асоційований з порожнім безліччю (тобто жоден приклад не потрапив в даний вузол), то він позначається як лист, і як рішення листа вибирається найбільш часто зустрічається клас у безпосереднього попередника даного листа.

З властивостей ентропії нам відомо, що максимально можливе значення ентропії досягається в тому випадку, коли всі його повідомлення різновірогідні. У нашому випадку, ентропія (3.4) досягає свого максимуму коли частота появи класів у прикладах множини T рівно ймовірна. Необхідно вибрати такий атрибут, щоб при розбитті по ньому один з класів мав найбільшу ймовірність появи. Це можливо в тому випадку, коли ентропія (3.4) матиме мінімальне значення і, відповідно, критерій (3.5) досягне свого максимуму.

У випадку з числовими атрибутами слід вибрати якийсь поріг, з яким повинні порівнюватися всі значення атрибута. Нехай числовий атрибут має кінцеве число значень. Позначимо їх {v1, v2 ... vn}. Попередньо відсортуємо всі значення. Тоді будь-яке значення, що лежить між vi та vi+1, ділить всі приклади на дві множини: ті, які лежать зліва від цього значення

{v1, v2 ... vi}, та ті, що праворуч {vi+1, vi+2 ... vn}. В якості порога можна вибрати середнє між значеннями vi та vi+1:

$$TH_i = \frac{v_i + v_{i+1}}{2} . \quad (3.6)$$

Таким чином суттєво спрощується завдання знаходження порога і приводиться до розгляду лише n-1 потенційних порогових значень TH1, TH2 ... THn-1.

Формули (3.3), (3.4) та (3.5) послідовно застосовуються до всіх потенційних порогових значень і серед них вибирається те, яке дає максимальне значення за критерієм (3.5). Далі це значення порівнюється зі значеннями критерію (3.5), підрахованими для решти атрибутів. Якщо з'ясується, що серед всіх атрибутів даний числовий атрибут має максимальне значення за критерієм (3.4), то в якості перевірки вибирається саме він.

Слід зазначити, що всі числові тести є бінарними, тобто ділять вузол дерева на дві гілки.

3.1.2 Класифікація нових прикладів

Отже, маємо дерево рішень для використання його у розпізнаванні нового об'єкту. Обхід дерева рішень починається з кореня дерева. На кожному внутрішньому вузлі перевіряється значення об'єкта Y за атрибутом, який відповідає перевірці в даному вузлі, і, залежно від отриманої відповіді, знаходиться відповідне розгалуження, і з цієї дузі рухаємося до вузла, що знаходиться на рівень нижче тощо. Обхід дерева закінчується як тільки зустрінеться вузол рішення, який і дає назва класу об'єкта Y.

Отже, перший крок зроблено. Ви можете легко отримати дерево рішень, реалізувавши описаний алгоритм. Та невеликим недоліком будуть дерева, якщо приклади представлені дуже великою кількістю атрибутів. Цей недолік можна усунути, застосувавши методику відсікання (pruning) гілок.

3.1.3 Покращений критерій розбиття

Критерій (3.5) має один недолік – йому краще підходять атрибути, які мають багато значень. Розглянемо гіпотетичну задачу медичної діагностики, де один з атрибутів ідентифікує особу пацієнта. Оскільки кожне значення цього атрибута унікальне, то при розбитті множини прикладів з цього атрибуту виходять підмножини, що містять тільки по одному прикладу. Так як всі ці множини "одно примірні", то і приклад відноситься, відповідно, до одного єдиного класу, тоді:

$$Info_X(T) = 0. \tag{3.7}$$

Це означає що критерій (3.5) приймає своє максимальне значення, та саме цей атрибут буде обрано алгоритмом. Однак, якщо розглянути проблему інакше – з точки зору здатності прогнозування побудованої моделі, то стає зрозуміла безкорисність такої моделі. Хоча на практиці не так часто зустрічаються подібні задачі, але необхідно передбачити і такі ситуації.

Проблема вирішується введенням деякої нормалізації. Нехай суть інформації повідомлення, що відноситься, наприклад, вказує не на клас, до якого приклад належить, а на вихід.

Тоді, за аналогією з визначенням *Info(T),* ми маємо вираз (3.8), що оцінює потенційну інформацію, отриману при розбитті множини *N* на n підмножини:

$$split\ info(X) = -\sum_{i=1}^{n} \frac{T_i}{T} \log_2 \left(\frac{T_i}{T}\right). \quad (3.8)$$

Розглянемо такий вираз(3.9). Нехай це є критерій вибору атрибуту:

$$gain\ ratio(X) = \frac{gain(X)}{split\ info(X)}. \quad (3.9)$$

У такому разі атрибут, що ідентифікує пацієнта, не буде високо оцінений критерієм (3.9). Нехай є k класів, тоді чисельник виразу (3.9) максимально буде дорівнює log2 (k) і нехай n - кількість прикладів в навчальній вибірці і одночасно кількість значень атрибуту, тоді знаменник максимально дорівнює log2 (n). Якщо припустити, що кількість прикладів свідомо більше кількості класів, то знаменник зростає швидше, ніж чисельник, і, відповідно, вираз буде мати невелике значення. Таким чином, ми можемо замінити критерій (3.5) на новий критерій (3.9), і знову ж вибрати той атрибут, який має максимальне значення за критерієм. Критерій (3.5) використовувався в алгоритмі ID3, критерій (3.9) введено в модифікованому алгоритмі C4.5.

Незважаючи на те, що було поліпшено критерій вибору атрибута для розбиття, алгоритм може створювати вузли і листи, що містять незначну кількість прикладів. Щоб уникнути цього, слід скористатися ще одним евристичним правилом: при розбитті множини T, принаймні дві підмножини повинні мати не менше заданого мінімальної кількості прикладів k (k> 1). Зазвичай воно дорівнює 2. У разі невиконання цього правила, подальше розбиття цієї множини припиняється, і відповідний вузол відзначається як лист.

Ймовірно, що при такому обмеженні може виникнути ситуація коли приклади асоційовані з вузлом відносяться до різних класів. В якості вирішення листа вибирається клас, який найбільш часто зустрічається у вузлі. Коли прикладів порівну з усіх класів, тоді рішення дає клас, щоь найчастіше зустрічається у безпосереднього предка даного листа.

Повертаючись до питання про вибір порога для числових атрибутів, можна ввести наступне доповнення: якщо мінімальне число прикладів у вузлі *k* тоді має сенс розглядати тільки такі значення $TH_1, TH_2 \ldots TH_{n-1}$, так як при розбитті по першим і останнім k-1 порогам в вузол потрапляє менше *k* прикладів.

3.1.4 Пропущені дані

Алгоритм побудови дерев рішень, розглянутий нами, припускає, що для атрибута, обраного в якості перевірки, існують всі значення, хоча явно це ніде не стверджувалося. Тобто для будь-якого прикладу з навчальної вибірки існує значення по цьому атрибуту.

Поки ми працюємо з синтетичними даними, особливих проблем не виникає, ми можемо "згенерувати" потрібні дані. Але як тільки ми звернемося до практичної сторони питання, то з'ясовується, що реальні дані далекі від ідеальних, і що часто зустрічаються припущення, що суперечать і аномальні дані.

Перше рішення, яке лежить на поверхні - це не враховувати приклади з пропущеними значеннями. Слід підкреслити, що вкрай небажано відкидати весь приклад тільки тому, що по одному з атрибутів пропущено значення, оскільки є ризик втратити багато корисної інформації.

Тоді необхідно виробити процедуру роботи з пропущеними даними. Нехай *T* - безліч навчальних прикладів,а *X* - перевірка по деякому атрибуту *A*.

Позначимо через *U* кількість невизначених значень атрибуту *A*. Змінимо формули (3) і (4) таким чином, щоб враховувати лише ті приклади, у яких існують значення по атрибуту *A*:

$$Info(T) = -\sum_{j=1}^{k} \frac{freq(C_j,T)}{|T|-U} * \log_2\left(\frac{freq(C_j,T)}{|T|-U}\right), \quad (3.10)$$

$$Info(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|-U} * Info(T_i). \quad (3.11)$$

У цьому випадку при підрахунку *freq(Cj,T)* враховуються тільки приклади з існуючими значеннями атрибута *A*.

Тоді критерій (3.5) можна переписати так:

$$Gain(X) = \frac{|T|-U}{|T|}(Info(T) - Info_X(T)). \quad (3.12)$$

Подібним чином змінюється і критерій (3.9). Якщо перевірка має *n* вихідних значень, то критерій (3.9) вважається таким, як у випадку, коли вихідна безліч розділена на *n+1* підмножин.

Нехай тепер перевірка *X* з вихідними значеннями O_1, O_2 ... O_n обрано на основі модифікованого критерію (3.11).

Треба вирішити, що робити з пропущеними даними. Якщо приклад з множини *T* з відомим виходом O_i асоційований з підмножиною T_i, ймовірність того, що приклад з безлічі T_i дорівнює 1. Нехай тоді кожен приклад з підмножини Ті має вагу, який вказує ймовірність того, що приклад належить T_i.

Якщо приклад має значення по атрибуту *A*, тоді вага дорівнює 1, в іншому випадку приклад асоціюється з усіма множинами $T_1, T_2 \ldots T_n$, з відповідними вагами:

$$\frac{|T_1|}{|T|-U}, \frac{|T_2|}{|T|-U}, \ldots, \frac{|T_x|}{|T|-U}, \tag{3.13}$$

$$\sum_{i=1}^{x} \frac{|T_i|}{|T|-U} = 1. \tag{3.14}$$

Коротко цей підхід можна сформулювати таким чином: передбачається, що пропущені значення по атрибуту розподілені із вірогідністю пропорційно частоті появи існуючих значень.

3.1.5 Класифікація нових прикладів

Така ж методика застосовується, коли дерево використовується для класифікації нових прикладів. Якщо на якомусь вузлі дерева при виконанні перевірки з'ясовується, що значення відповідного атрибута, що класифікується для даного прикладу, пропущено, то алгоритм досліджує всі можливі шляхи вниз по дереву і визначає з якою ймовірністю приклад відноситься до різних класів. У цьому випадку, "класифікація" - це скоріше розподіл класів. Як тільки розподіл класів встановлено, то клас, що має найбільшу ймовірність появи, вибирається в якості відповіді дерева рішень.

Слід зазначити, що крім підходу, використаного в описуваному алгоритмі, застосовуються й інші методики. Зрештою, успіх від використання того чи іншого методу роботи з пропущеними даними, безпосередньо залежить як від предметної області, так і самих даних.

Дерева рішень мають свої переваги та недоліки. До переваг відносять:

– простий в розумінні та інтерпретації. Люди здатні інтерпретувати результати моделі дерева прийняття рішень після короткого пояснення;

– не вимагає підготовки даних. Інші техніки вимагають нормалізації даних, додавання фіктивних змінних, а також видалення пропущених даних;

– дозволяє оцінити модель за допомогою статистичних тестів. Це дає можливість оцінити надійність моделі;

– є надійним методом. Метод добре працює навіть в тому випадку, якщо були порушені початкові припущення, включені в модель.

До недоліків у свою чергу відносять наступне:

– ті, хто вивчає метод дерева прийняття рішень, можуть створювати занадто складні конструкції, які недостатньо повно представляють дані;

– існують концепти, які складно зрозуміти з моделі, так як модель описує їх складним шляхом.

3.2 Опис технологій

Для розробки інформаційної системи тестування веб додатків мною була обрана мова програмування Java. На мій погляд дана мова програмування має багато зручних інструментів для досягнення поставленої задачі, а до того ж дана мова є кроссплатформенною.

Задля реалізації програми на мові програмування Java була обрана платформа NetBeans.

Для того, щоб працювати із файлами мною був використаний пакет java.io, що має набір методів для роботи із потоками вводу та виводу даних. А саме були використані потоки:

– file – для створення нового файлу та запису до вже існуючого;

– fileInputStream – поток вводу даних, що дозволяє считати

інформацію із файлами;

– inputStreamReader – поток, який є мостом від потоків байтів до символьних потоків, тобто считує байти і декодує їх у символи;

– bufferedReader – поток, що дозволяє читати інформацію та занести її до буферу;

– printWriter – поток, що використовується для форматированного виведення даних до файлу.

3.3 Опис програми

Для будь-якої програми є вхідні та вихідні дани. У якості вхідних даних у моїй програмі використовується сайт Alexa.com, де зібрані найбільш популярні сайти. У даному випадку «найбільш популярні» означає не лише те, що на них дуже часто заходять люди, а й те, що ці сайти є якісними. Тож для своїє роботи я обрала кілька сайтів із верху рейтинга та кілька сайтів із низу рейтинга аби результати можна було порівняти та зрозуміти чи справедливо сайти займають свої місця у рейтингу.

Наступним кроком є проведення навантажувального тестування. У результаті ми отримаємо багато різних даних. Далі по кожному з наборів даних ми рахуємо показники. Кожен із наборів таких показників заноситься до файлу.

Після цього відбувається побудова класифікатора алгоритмами машинного навчання. Для цієї задачі використовується клас алгоритмів дерев рішень. Зокрема було використано алгоритм C4.5. У результаті ми отримуємо правила оцінки сайту, по яким можемо відразу оцінити той чи інший сайт.

Після цього можна застосовувати ці правила оцінки. Тобто хтось, хто буде застосовувати правила, повинен провести підготовчі роботи. А саме

провести навантажувальне тестування та отримати набір даних. Порахувати набір показників та підставити файл із ними до класифікатору.

3.4 Аналіз результатів

Програма, що розроблялася під час виконання дипломного проекту, рахує параметри, що необхідні для побудови правил оцінки. У якості вхідних параметрів беруться дані, що були зібрані під час навантажувального тестування, а саме кількість користувачів та час завантаження сторінки. Рахується середній час, мінімальний та максимальний час, кореляція та швидкість зросту. Як виглядає загальний файл із розрахованими показниками для побудови правил представлений на рис. 3.1.

User	Amazon	Niteflirt.c	Paypal.co	American	Intelli-sho	Olx.ua	Rozetka.c	www.scie	bad1	bad2	bad3
10	0,030	0,028	0,018	0,013	0,050	0,083	0,018	0,013	0,283	0,359	0,366
20	0,024	0,045	0,015	0,015	0,024	0,045	0,015	0,045	0,145	0,238	0,246
30	0,025	0,050	0,020	0,015	0,050	0,070	0,017	0,015	0,170	0,179	0,188
50	0,035	0,056	0,016	0,016	0,035	0,056	0,016	0,026	0,260	0,286	0,295
100	0,031	0,029	0,020	0,013	0,053	0,089	0,019	0,013	0,204	0,229	0,235
150	0,031	0,049	0,016	0,015	0,029	0,046	0,015	0,045	0,199	0,230	0,233
200	0,030	0,054	0,021	0,019	0,050	0,077	0,018	0,029	0,283	0,329	0,338
300	0,031	0,057	0,016	0,016	0,056	0,076	0,016	0,015	0,245	0,266	0,274
400	0,039	0,056	0,020	0,019	0,059	0,081	0,019	0,030	0,190	0,260	0,268
500	0,031	0,033	0,017	0,013	0,051	0,088	0,016	0,014	0,260	0,313	0,320
600	0,031	0,063	0,017	0,019	0,059	0,077	0,017	0,030	0,277	0,289	0,293
700	0,030	0,058	0,020	0,019	0,054	0,078	0,020	0,030	0,373	0,438	0,444
1000	0,031	0,061	0,017	0,019	0,051	0,079	0,017	0,030	0,383	0,481	0,487
1500	0,027	0,050	0,020	0,016	0,059	0,075	0,017	0,015	0,390	0,488	0,497
2000	0,041	0,064	0,018	0,015	0,062	0,086	0,018	0,016	0,595	0,604	0,607
Среднее время	0,031	0,050	0,018	0,016	0,049	0,074	0,017	0,024	0,284	0,333	0,339
Минимальное	0,024	0,028	0,015	0,013	0,024	0,045	0,015	0,013	0,145	0,179	0,188
Максимальное	0,041	0,064	0,021	0,019	0,062	0,089	0,020	0,045	0,595	0,604	0,607
Корреляция	0,422	0,470	0,164	0,229	0,560	0,376	0,221	-0,230	0,900	0,891	0,889
Скорость роста	0,000	0,002	0,000	0,000	0,002	0,001	0,000	0,000	0,019	0,020	0,020

Рисунок 3.1 – Файл із розрахованими показниками

Після того, як усі необхідні показники розраховані ми можемо

підставити ці дані у програму Deductor.

На підставі отриманих даних програма побудує дерево рішень, де будуть розписані усі перевірки, що потрібно буде провести для побудови правил оцінки. Дерево рішень представлено на рис. 3.2.

Які з порахованих показників потрібно враховувати при оцінці показано на вкладниці значимості атрибутів. Так як дана вибірка невелика для оцінки потрібно перевірити лише два показники середній час та кореляцію. Приклад значимості атрибутів показаний на рис. 3.3.

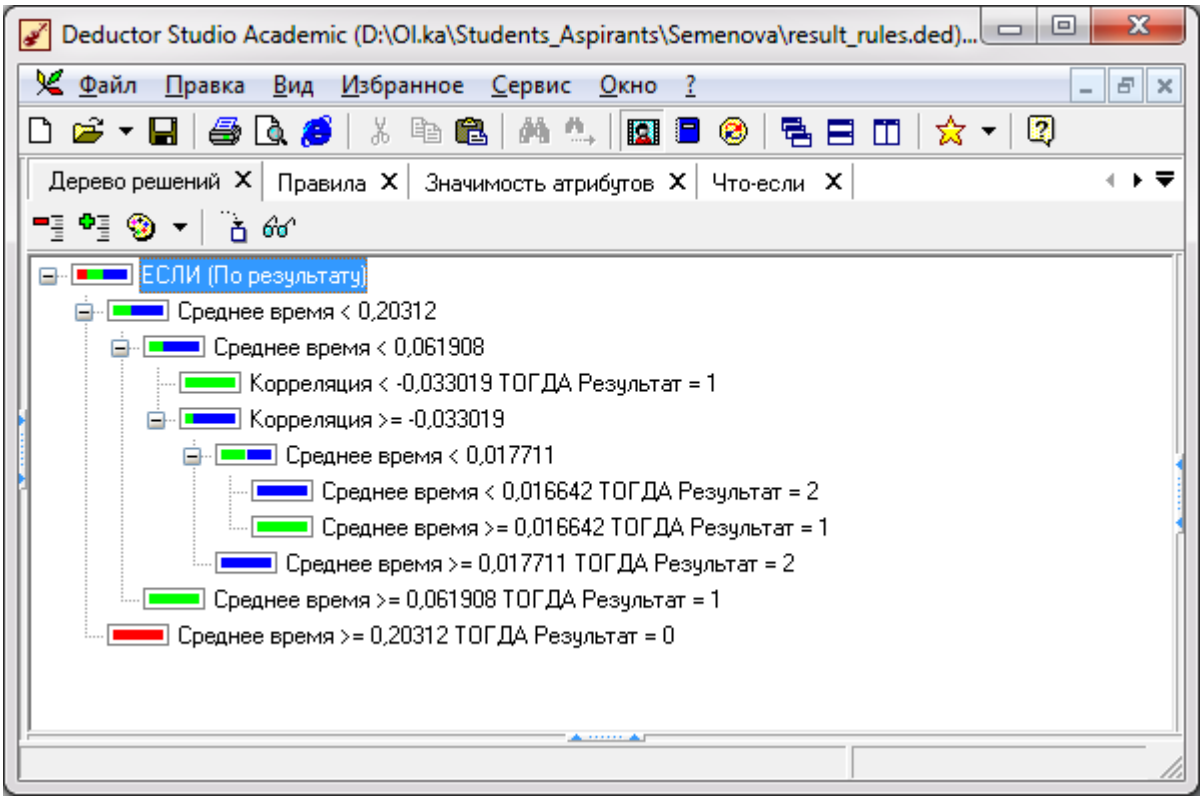


Рисунок 3.2 – Дерево рішень у Deductor

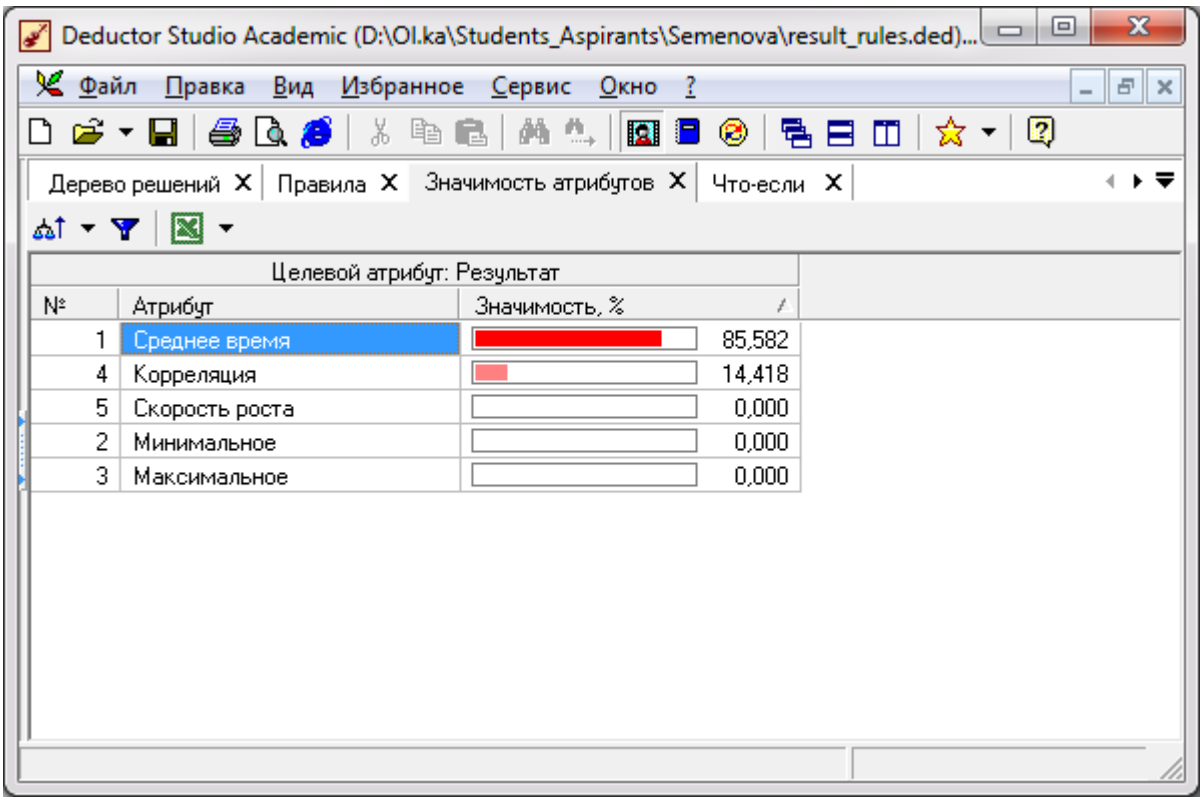


Рисунок 3.3 – Значимість атрибутів у Deductor

У результаті програма побудує правила для оцінки, що представлені у таблиці 3.1.

Таблиця 3.1 – Правила оцінки у програмі Deductor

№	Умова	Слідство (Результат)	Підтримка		Достовірність	
			%	Кіл-ть	%	Кіл-ть
1	Среднее время < 0,20312 **И** Среднее время < 0,061908 **И** Корреляция < -0,033019	1	10,00	1	100,00	1
2	Среднее время < 0,20312 **И** Среднее время < 0,061908 **И** Корреляция >= -0,033019 **И** Среднее время < 0,017711 **И** Среднее время < 0,016642	2	10,00	1	100,00	1
3	Среднее время < 0,20312 **И** Среднее время < 0,061908 **И** Корреляция >= -0,033019 **И** Среднее время < 0,017711 **И** Среднее время >= 0,016642	1	10,00	1	100,00	1
4	Среднее время < 0,20312 **И** Среднее время < 0,061908 **И** Корреляция >= -0,033019 **И** Среднее время >= 0,017711	2	40,00	4	100,00	4
5	Среднее время < 0,20312 **И** Среднее время >= 0,061908	1	10,00	1	100,00	1
6	Среднее время >= 0,20312	0	20,00	2	100,00	2

У стовпчики «Умова» розписані усі умови, які потрібно перевірити для оцінки. «Слідство» показує саму оцінку. Нуль відповідає найгіршій оцінці, а два – найкращій. На прикладі даної вибірки ми бачимо, що програма дає сто відсоткову достовірність.

Тож у результаті ми маємо оцінки, що представлені на рис. 3.4.

	Среднее время	Минимальное	Максимальное	Корреляция	Скорость роста	Результат
Amazon	0,031	0,024	0,041	0,422	0,000	2
Niteflirt.com	0,050	0,028	0,064	0,470	0,002	2
Paypal.com	0,018	0,015	0,021	0,164	0,000	2
Americanexpress.com	0,016	0,013	0,019	0,229	0,000	2
Intelli-shop.com	0,049	0,024	0,062	0,560	0,002	2
Olx.ua	0,074	0,045	0,089	0,376	0,001	1
Rozetka.com.ua	0,017	0,015	0,020	0,221	0,000	1
www.sciencedaily.com	0,024	0,013	0,045	-0,230	0,000	1
bad1	0,284	0,145	0,595	0,900	0,019	0
bad2	0,333	0,179	0,604	0,891	0,020	0
bad3	0,339	0,188	0,607	0,889	0,020	0

Рисунок 3.4 – Оцінки сайтів

4 ОХОРОНА ПРАЦІ

4.1 Аналіз потенційних небезпечних і шкідливих виробничих чинників проектованого об’єкту, що мають вплив на персонал

У даному дипломному проекті розробляється програмне забезпечення.

Розроблене програмне забезпечення орієнтоване на роботу з персональним комп'ютером. Експлуатовані для вирішення внутрішньовиробничих завдань ПЕОМ типу IBM PC мають наступні характеристики:

споживана потужність	220 Вт;
робоча напруга	220 В;
напруга джерел живлення	+12 В; - 12 В; +5 В;
робоча частота	50 Гц.

Виходячи з приведених характеристик, вочевидь, що для людини існує небезпека поразки електричним струмом, унаслідок недбалого поводження з комп'ютером і порушення правил експлуатації, залишення частин ПЕОМ, що знаходяться під напругою, відкритими або знятих для ремонту вузлів.

Відповідно до [12] до легкої фізичної роботи відносяться всі види діяльності, виконувані сидячи і ті, що не потребують фізичної напруги. Робота користувача ПК відноситься до категорії 1а.

При роботі на ПЕОМ користувач піддається ряду потенційних небезпек. Унаслідок недотримання правил техніки безпеки при роботі з машиною(невиконання огляду відкритих частин ПЕОМ, що знаходяться під напругою або знятих для ремонту вузлів) для користувача існує небезпека поразки електричним струмом.

Джерелами підвищеної небезпеки можуть служити наступні елементи:

– розподільний щит;

– джерела живлення;

– блоки ПЕОМ і друку, що знаходяться в ремонті.

Ще одна проблема полягає у тому, що спектр випромінювання комп'ютерного монітора включає рентгенівську, ультрафіолетову і інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівського проміння мала, оскільки цей вид випромінювання поглинається речовиною екрану. Проте велику увагу слід приділяти біологічним ефектам низькочастотних електромагнітних полів(аж до порушення ДНК).

Відповідно до [13], при обслуговуванні ПЕОМ мають місце фізичні і психофізичні небезпечні, а також шкідливі виробничі чинники:

– підвищене значення напруги в електричному ланцюзі, замикання якої може відбутися через тіло людини;

– підвищений рівень статичної електрики;

– підвищений рівень електромагнітних випромінювань;

– підвищена або знижена температура повітря робочої зони;

– підвищений або знижений рух повітря;

– підвищена або знижена вологість повітря;

– відсутність або недостатність природного світла;

– підвищена пульсація світлового потоку;

– недостатня освітленість робочого місця;

– підвищений рівень шуму на робочому місці;

– розумове перенапруження;

– емоційні навантаження;

– монотонність праці.

4.2 Заходи щодо техніки безпеки

Основним небезпечним чинником при роботі з ЕОМ є небезпека поразки людини електричним струмом, яка посилюється тим, що органи чуття людини не можуть на відстані знайти наявності електричної напруги на устаткуванні.

Проходячи через тіло людини, електричний струм чинить на нього складну дію, що є сукупністю термічної(нагрів тканин і біологічних середовищ), електролітичної(розкладання крові і плазми) і біологічної(роздратування і збудження нервових волокон і інших органів тканин організму) дій.

Тяжкість поразки людини електричним струмом залежить від цілого ряду чинників:

– значення сили струму;

– електричного опору тіла людини і тривалості протікання через нього струму;

– роду і частоти струму;

– індивідуальних властивостей людини і навколишнього середовища.

Розроблений дипломний проект передбачає наступні технічні способи і засоби, що застерігають людину від ураження електричним струмом [14]:

– заземлення електроустановок;

– занулення;

– захисне відключення;

– електричне розділення ятерів;

– використовування малої напруги;

– ізоляція частин, що проводять струм;

– огорожа електроустановок.

Занулення зменшує напругу дотику і обмежує година, протягом якого людина, ткнувшись до корпусу, може потрапити під дію напруги.

Струм однофазного короткого замикання визначається по наближеній формулі:

$$I_K = \frac{\text{Uф}}{\text{Zп} + \frac{\text{Zт}}{3}}\ , \qquad (4.1)$$

де Uф - номінальна фазна напруга мережі, В;

Zп - повний опір петлі, створене фазними і нульовими дротами, Ом;

Z_T - повний опір струму короткого замикання на корпус, Ом.

Згідно таблиці 4 [15]: $Z_T/3 = 0{,}1$ Ом.

Для провідників і жил кабелю для розрахунку повного опору петлі використовуємо формулу(4.2.) :

$$\text{Zп} = \sqrt{\text{Rп}^2 + \text{Xп}^2}\ , \qquad (4.2)$$

де Rп = Rф + Ro - сумарний активний опір фазного Rф і нульового Ro дротів, Ом;

Xп - індуктивний опір паяння дротів, Ом.

Перетин 1 км мідного дроту S = 2.5 мм, тоді згідно таблицям 5 і 6 [18], має такий опір:

Xп = 0,11 Ом;

Rф = 7,55 Ом;

Ro = 7,55 Ом.

Отже, Rп = 7,55 + 7,55 = 15,1 Ом.

Тоді по формулі (4.2) знаходимо повний опір петлі:

$$Zп = \sqrt{15,1^2 + 0,11^2} \approx 15,1 \text{ (Ом)}.$$

Струм однофазного короткого замикання рівний:

$$Iк = \frac{220}{15,1 + 0,1} = 14,47 \text{ (А)}.$$

Дія плавкої вставки на ПЕОМ забезпечується, якщо виконується співвідношення:

$$Iк \geq k * Iн, \quad (4.3)$$

де Iн - номінальний струм спрацьовування плавкої вставки, А;

k - коефіцієнт кратності нелінійного струму Iн, А.

Коефіцієнт кратності нелінійного струму Iн розраховується по формулі (4.4.) :

$$Iн = P / U, \quad (4.4)$$

де P = 220 Вт - споживана потужність;

U = 220 В - робоча напруга;

k = 3 А - для плавких вставок.

Отже, Iн = 220 / 220 = 1 А.

Підставивши значення у вираз (4.3), одержимо:

$$14,47 > 3*1.$$

Таким чином, доведено, що апарат забезпечить спрацьовування(і захист) при підвищенні номінального струму.

4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці

Вимоги до виробничих приміщень встановлюються [14], СНіП, відповідними ГОСТами і ОСТамі з урахуванням небезпечних і шкідливих чинників, що утворюються в процесі експлуатації електроустаткування.

Підвищення працездатності людини і збереження її здоров'я забезпечується стабільними метеорологічними умовами. Мікроклімат виробничих приміщень [17] визначається діючими на організм людини поєднаннями температури, вологості і швидкості руху повітря, а також температури навколишніх поверхонь. Значне коливання параметрів мікроклімату приводити до порушення систем кровообігу, нервової і потовидільної, що може викликати підвищення або пониження температури тіла, слабкість, запаморочення і навіть непритомність.

Відповідно до [18] встановлюють оптимальну і допустиму температуру, відносну вологість і швидкість руху повітря в робочій зоні . За відсутності надмірного тепла, вологи, шкідливих речовин в приміщенні досить природної вентиляції.

У приміщенні для виконання робіт операторського типу(категорія 1а), пов'язаних з нервово-емоційною напругою, проектом передбачається дотримання наступних нормованих величин параметрів мікроклімату (табл. 4.1).

Таблиця 4.1 - Санітарні норми мікроклімату робочої зони приміщень для робіт категорії 1а.

Пора року	Температура, С	Відносна вологість, %	Швидкість руху повітря, м/с
Холодна	22…24	40…60	0,1
Тепло	23…25	40…60	0,1

У приміщенні, де знаходиться ПЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції(з пристроєм вентиляційних каналів в перекриттях будівлі і вертикальних шахт) й установленого промислового кондиціонера фірми Mitsubishi, який дозволяє вирішити переважну більшість завдань по створінню та підтримці необхідних параметрів повітряного середовища. Цей метод забезпечує приток потрібної кількості свіжого повітря, визначеного в СНіП (30 $м^3$ в годину на одного працівника).

Шум на виробництві має шкідливу дію на організм людини. Стомлення операторів через шум збільшує число помилок при роботі, призводить до виникнення травм. Для оператора ПЕОМ джерелом шуму є робота принтера. Щоб усунути це джерело шуму, використовують наступні методи. При покупці принтера слід вибирати найбільш шумозахисні матричні принтери або з великою швидкістю роботи(струменеві, лазерні). Рекомендується принтер поміщати в найбільш віддалене місце від персоналу, або застосувати звукоізоляцію та звукопоглинання(під принтер підкладають демпфуючі підкладки з пористих звукопоглинальних матеріалів з листів тонкої повсті, поролону, пеноплену).

При роботі на ПЕОМ, проектом передбачені наступні методи захисту від електромагнітного випромінювання : обмеження часом, відстанню, властивостями екрану.

Обмеження годині роботи на ПЕОМ складає 3,5-4,5 години. Захист відстанню передбачає розміщення монітора на відстані 0,4-0,5 м від оператора. Передбачений монітор 20" TFT, Samsung 2043BW відповідає вимогам стандарту [19].

Стандарт [19] пред’являє жорсткі вимоги в таких областях: ергономіка(фізична, візуальна і зручність користування), енергія, випромінювання(електричних і магнітних полів), навколишнє середовище і екологія, а також пожежна та електрична безпека, які відповідають всім

вимогам [20].

Для зниження стомлюваності та підвищення продуктивності праці обслуговуючого персоналу в колірній композиції інтер'єру приміщень для ПЕОМ дипломним проектом пропонується використовувати спокійні колірні поєднання і покриття, що не дають відблисків.

У проекті передбачається використання сумісного освітлення. У світлий час доби приміщення освітлюватиметься через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Як штучне освітлення необхідно використовувати штучне робоче загальне освітлення. Для загального освітлення необхідно використовувати люмінесцентні лампи. Вони володіють наступними перевагами: високою світловою віддачею, тривалим терміном служби, хоча мають і недоліки: високу пульсацію світлового потоку.

При експлуатації ПЕОМ виробляється зорова робота. Відповідно до [21] ця робота відноситься до розряду 5а. При цьому нормоване освітлення на робочому місці(Ен) при загальному освітленні рівна 200 лк.

Приміщення завдовжки 12 м, шириною 10 м, заввишки 4 м обладнується світильниками типу ЛП02П, оснащеними лампами типу ЛБ зі світловим потоком 3120 лм кожна.

Виконаємо розрахунок кількості світильників в робочому приміщенні завдовжки а=12 м, шириною b=10 м, заввишки з=4 м, використовуючи формулу (4.5) розрахунку штучного освітлення при горизонтальній робочій поверхні методом світлового потоку:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \qquad (4.5)$$

де F - світловий потік = 3120 лм;

E - максимально допустима освітленість робочих поверхонь = 200 лк;

S - площа підлоги = 120 м2;

Z - поправочний коефіцієнт світильника = 1,2;

k - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації світильників = 1,5;

n - кількість світильників;

U - коефіцієнт використовування освітлювальної установки = 0,6;

M - кількість ламп у світильнику =2.

З формули (4.5) виразимо n (4.6) і визначимо кількість світильників для даного приміщення:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \quad (4.6)$$

Отже, $n = (200 \cdot 120 \cdot 1{,}2 \cdot 1{,}5)/(3120 \cdot 0{,}6 \cdot 2) = 12$.

Виходячи з цього, рекомендується використовувати 12 світильників. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. 4.1.

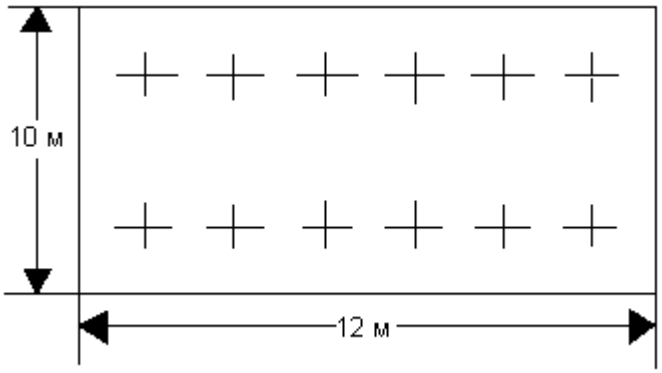


Рисунок 4.1 - Схема розташування світильників

4.4 Рекомендації по пожежній безпеці

Пожежі в приміщеннях, де встановлена обчислювальна техніка, представляють небезпеку для життя людини. Пожежі також пов’язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки, що у свою чергу спричиняє за собою порушення ходу технологічного процесу.

Пожежа може виникнути при наявності горючої речовини та внесення джерела запалювання в горюче середовище. Пальними матеріалами в приміщеннях, де розташовані ПЕОМ, є:

– поліамід - матеріал корпусу мікросхеми, горюча речовина, температура самозаймання аерогелю 420 С ;

– полівінілхлорид - ізоляційний матеріал, горюча речовина, температура запалювання 335 С, температура самозаймання 530 С, кількість енергії, що виділяється при згоранні - 18000 - 20700 кДж/кг;

– стеклотекстоліт ДЦ - матеріал друкарських плат, важкозаймистий матеріал, показник горючості 1.74, не схильний до температурного самозаймання;

– пластика кабельний №489 - матеріал ізоляції кабелю, горючий матеріал, показник горючості більш 2.1;

– деревина - будівельний і обробний матеріал, матеріал з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, теплота згорання 18731 - 20853 кДж/кг, температура запалювання 399 З, схильна до самозаймання [23].

Згідно [24] приміщення відносяться до категорії В(пожежовибухонебезпечним) і згідно правилам побудови електроустановок простір усередині приміщення відноситься до вогненебезпечної зони класу П - IIа (зони, розташовані в приміщеннях, в яких зберігаються тверді горючі

речовини).

Потенційними джерелами запалення при роботі ПЕОМ є:

– іскри при замиканні і розмиканні ланцюгів;

– іскри і дуги коротких замикань;

– перегріви від тривалого перевантаження і наявності перехідного опору.

Продуктами згорання, що виділяються при пожежі, є : оксид вуглецю, сірчистий газ, оксид азоту, синильна кислота, акропеін, фосген, хлор та ін. При горінні пластмас, окрім звичайних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідрідні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол та ін., що шкідливо впливають на організм людини.

Для захисту персоналу від дії небезпечних і шкідливих чинників пожежі проектом передбачається застосування промислового протигаза з коробкою марки В(жовта).

Пожежна безпека об’єктів народного господарства регламентується [18] і забезпечується системами запобігання пожежам і протипожежному захисту. Для успішного гасіння пожеж вирішальне значення має швидке виявлення пожежі і своєчасний виклик пожежних підрозділів до місця пожежі.

Зменшити горюче навантаження не представляється можливим, тому проектом передбачається застосувати наступні способи і їх комбінації для запобігання утворенню(внесення) джерел запалення :

– застосування устаткування, що задовольняє вимогам електростатичної безпеки;

– застосування в конструкції швидкодіючих засобів захисного відключення можливих джерел запалення;

– виключення можливості появи іскрового заряду статичної електрики в горючому середовищі з енергією, рівної і вище мінімальної енергії запалення;

– підтримка температури нагріву поверхні машин, механізмів, устаткування, пристроїв, речовин і матеріалів, які можуть увійти до контакту з пальним середовищем, нижче гранично допустимої, становить 80% якнайменшої температури самозаймання пального.

– заміна небезпечних технологічних операцій більш безпечними;

– ізольоване розташування небезпечних технологічних установок і устаткування;

– зменшення кількості пальних і вибухонебезпечних речовин, що знаходяться у виробничих приміщеннях;

– запобігання можливості утворення пальних сумішей на лінії, вентиляційних системах і ін.;

– механізація, автоматизація та справність(потокова) виробництва;

– суворе дотримання стандартів і точне виконання встановленого технологічного режиму;

– запобігання можливості появи в небезпечних місцях джерел запалення;

– запобігання розповсюдженню пожеж і вибухів;

– використовування устаткування і пристроїв, при роботі яких не виникає джерел запалення;

– виконання вимог сумісного зберігання речовин і матеріалів;

– наявність громовідводу;

– організація автоматичного контролю параметрів, що визначають джерела запалення;

– ліквідація можливості самозаймання речовин і матеріалів .

– Для запобігання пожежі в обчислювальних центрах проектом пропонується виконання наступних вимог :

– електроживлення ЕОМ повинно мати автоматичне блокування відключення електроенергії на випадок зупинки системи охолоджування і кондиціонування;

– система вентиляції обчислювальних центрів повинна бути обладнана блокуючими пристроями, що забезпечують її відключення на випадок пожежі;

– робочі місця повинні бути оснащені пожежними щитами, сигналізацією, засобами для сповіщення про пожежну небезпеку (телефонами), медичними аптечками для надання першої медичної допомоги, розробленим планом евакуації.

Для зниження пожежної небезпеки в приміщеннях використовуються первинні засоби гасіння пожеж, а також система автоматичної пожежної сигналізації, яка дозволяє знайти початкову стадію загоряння, швидко і точно оповістити службу пожежної охорони про час і місце виникнення пожежі.

Відповідно до [24] приміщення категорії В підлягають устаткуванню системами автоматичної пожежної сигналізації. Проектом передбачається застосування датчика типу ІДФ - 1(димовий фотоелектричний датчик), оскільки специфікою пожеж обчислювальної техніки і радіоапаратури є, в першу чергу, виділення диму, а потім - підвищення температури.

При виникненні пожежі в робочому приміщенні обслуговуючий персонал зобов'язаний негайно вжити заходи по ліквідації пожежі. Для ліквідації пожежі використовують вогнегасники (хімічно-пінні, пінні для повітря ОП-5, ОП-6, ОП-9, вуглекислотні ОУ-5), пісок, пожежний інвентар(сокири, ломи, багри, шерстяну або азбестову ковдри) [20]. Як засіб індивідуального захисту проектом передбачається використовування промислового протигаза з маскою, фільтруючої коробки В.

В якості організаційно-технічних заходів рекомендується проводити навчання робочого персоналу правилам пожежної безпеки.

У розділі «Охорона праці» виконано аналіз потенційних небезпек при роботі із засобами обчислювальної техніки і механізмами, розроблені заходи щодо техніки безпеки, заходи, які забезпечують виробничу санітарію і гігієну

ВИСНОВКИ

Таким чином, у результаті проведеного аналізу проблеми недостатньої надійності продукції можна зазначити надважливу значимість навантажувального тестування. Даний тип тестування важливий не тільки для веб-додатків, а й взагалі для будь-якого програмного забезпечення.

Під час виконання дипломного проекту мною були детально розглянуті різновиди навантажувального тестування та програми для проведення цього типу тестування.

При реалізації даного проекту я спробував розробити класифікатор за допомогою алгоритмів машинного навчання для побудови правил оцінки сайтів. Для досягнення цілі використовувався клас алгоритмів дерев рішень, а саме алгоритм C4.5.

При виконанні розділу «Охорона праці» були розглянуті питання взаємодії системи «людина - машина-середовище», проведено аналіз умов праці, визначені небезпечні та шкідливі виробничі фактори, що мають місце в даному офісі, і розглянуто їх вплив на людину. Розроблені організаційні і технічні заходи, що зменшують або виключають вплив небезпечних та шкідливих виробничих факторів на людину.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1) Азгальдов Г. Г. Теорія і практика оцінки якості товарів [Текст] — М.: Экономика, 1982. — 256 с.

2) Калбертсон Р. Швидке тестування [Текст]: учеб. пособие /Калбертсон Р., Браун К., Кобб Г. — М.: «Вильямс», 2002. — 374 с. — ISBN 5-8459-0336-X.

3) Навантажувальне тестування[Электронный ресурс] / Википедия. Свободная энциклопедия. – Режим доступа: www/ URL : https://ru.wikipedia.org/wiki/Load_testing – 10.05.2018 р. – Загл. з екрану.

4) IBM Rational Performance Tester[Электронный ресурс] / Стаття. – Режим доступа: www/ URL : http://easy-code.com.ua/2012/09/navantazhuvalne-testuvannya-web-dodatkiv-za-dopomogoyu-ibm-rational-performance-tester-chastina-1-oglyad-funkcij-zasobiv-i-zvitiv-rizne-programuvannya-statti/ – 11.05.2018 р. – Загл. з екрану

5) Pylot [Электронный ресурс] / Стаття. – Режим доступа: www/ URL : http://iqa.com.ua/soft/load/pylot_web_load_testing_utility – 11.05.2018 р. – Загл. з екрану

6) Go[Электронный ресурс] / Стаття. – Режим доступа: www/ URL : http://habrahabr.ru/post/187212/ – 11.05.2018 р. – Загл. з екрану.

7) HP LoadRunner[Электронный ресурс] / Стаття. – Режим доступа: www/ URL : https://ru.wikipedia.org/wiki/HP_LoadRunner – 11.05.2018 р. – Загл. з екрану.

8) Siege [Электронный ресурс] / Стаття. – Режим доступа: www/ URL : http://www.fight.org.ua/publications/Siege--utilita-dlya-nagruzochnogo-testirovaniya-vebserverov.html – 11.05.2018 р. – Загл. з екрану.

9) Gregory, L.D. HTML5 Differences from HTML 4 [Текст] / L.D. Gregory - Working Draft. World Wide Web Consortium 2011. - 134 с.

10) К. Дари, Э. Баланеску. PHP и MySQL, Beginning PHP and MySQL E-Commerce: From Novice to Professional. [Текст] / К. Дари, Э. Баланеску, М.: «Вильямс», 2010. – 605 с.

11) С. Суэринг, Т. Конверс, Д. Парк. PHP и MySQL [Текст] / пер. с англ./ Стив Суэринг, Тим Конверс, Джойс Парк - 2001. - 704 с.

12) Державний стандарт України. ГОСТ 12.1.005-88. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

13) Державний стандарт України. ГОСТ 12.1.005-88. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

14) Державний стандарт України. ГОСТ 12.0.003-74 Опасные и вредные производственные факторы. Классификация.

15) Нормативно-правові акти з охорони праці. НПАОП 40.1-1.21-98. Правила безпечної експлуатації електроустановок споживачів

16) Державний стандарт України. ГОСТ 12.1.009-76. ССБТ. Электробезопасность. Термины и определения.

17) Державні санітарні норми України. ДСП 173-96. Державні санітарні правила планування та забудови населених пунктів.

18) Державні санітарні норми України. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень.

19) Державний стандарт України. ГОСТ 12.1.005-88. Система стандартов безопасности труда. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

20) TCO’ 07 Certified Displays. © 2007 Copyright TCO Development AB

21) Державні санітарні норми і правила. ДСанПіН 3.3.2.007-98, Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

22) Державні будівельні норми ДБН В.2.5-28-2006. Природне і штучне освітлення

23) Державний стандарт України. ГОСТ 12.1.044-89 Система стандартов безопасности труда. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения.

24) Нормативні акти пожежної безпеки. НАПБ Б.03.002-2007. Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою.

25) Державний стандарт України. ГОСТ 12.1.004-91. "Система стандартов безопасности труда. Пожарная безопасность. Общие требования".

26) Нормативні акти пожежної безпеки. НАПБ А.01.001-2014 “Правила пожежної безпеки в Україні”

27) Нормативні акти пожежної безпеки. НАПБ Б.03.001-2004. Про затвердження Типових норм належності вогнегасників.

Додаток А.

Електронні плакати

Дипломна робота бакалавра

Інформаційна система тестування веб додатків

Виконав:
ст. гр. КІ-143
Трубніков В.М.

Керівник проекту::
Недзельський Д.О.

Актуальність

- Буває так, що при вході на якийсь веб-сайт сторінка завантажується дуже довго
- Скільки ще людей пробують зараз відвідати сайт?

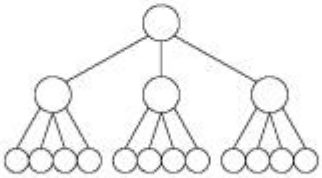
Постановка задачі

- 1) провести аналіз підходів до тестування
- 2) розробити вимоги до продуктивності веб-додатків
- 3) розробити програму, яка буде проводити аналіз даних
- 4) розробити правила оцінки сайтів

Загальна схема рішення

Дерево рішень

- Задача полягає в побудові ієрархічної класифікаційної моделі у вигляді дерева з безлічі прикладів



- Простий в розумінні та інтерпретації

Програмна реалізація

- Мова програмування: Java
- Вхідні дані: результати тестування навантаження
- Вихідні дані: правила оцінки сайтів
- Результат: оцінка сайту

Аналіз результатів

User	Amazon	Nifeflirt.c	Paypal.com	American	Intelli-shi	Olx.ua	Rozetka.u	www.sole	bad1	bad2	bad3
10	0,030	0,028	0,018	0,013	0,050	0,083	0,018	0,013	0,283	0,019	0,166
20	0,026	0,045	0,015	0,015	0,024	0,045	0,015	0,045	0,145	0,238	0,246
30	0,025	0,050	0,020	0,015	0,056	0,070	0,017	0,015	0,170	0,179	0,180
50	0,025	0,056	0,016	0,016	0,035	0,056	0,016	0,026	0,260	0,286	0,295
100	0,031	0,029	0,020	0,013	0,053	0,089	0,019	0,013	0,204	0,229	0,235
150	0,031	0,049	0,016	0,015	0,025	0,046	0,015	0,045	0,199	0,230	0,235
200	0,030	0,054	0,021	0,019	0,050	0,077	0,018	0,029	0,283	0,329	0,336
300	0,031	0,057	0,016	0,016	0,036	0,079	0,016	0,015	0,245	0,266	0,276
400	0,029	0,056	0,020	0,019	0,038	0,081	0,019	0,030	0,190	0,260	0,268
500	0,031	0,053	0,017	0,013	0,031	0,088	0,016	0,016	0,260	0,313	0,320
600	0,031	0,053	0,017	0,019	0,038	0,077	0,017	0,030	0,277	0,288	0,293
700	0,030	0,058	0,020	0,019	0,054	0,070	0,020	0,030	0,371	0,438	0,444
1000	0,031	0,051	0,017	0,019	0,051	0,079	0,017	0,030	0,383	0,461	0,467
1500	0,027	0,050	0,020	0,016	0,039	0,075	0,017	0,015	0,390	0,488	0,497
2000	0,041	0,064	0,018	0,013	0,062	0,086	0,018	0,038	0,595	0,604	0,607
Среднее время	0,031	0,050	0,018	0,016	0,049	0,076	0,017	0,026	0,284	0,333	0,339
Минимальное	0,026	0,028	0,015	0,013	0,024	0,045	0,015	0,013	0,145	0,179	0,180
Максимальное	0,041	0,064	0,021	0,019	0,062	0,089	0,020	0,045	0,595	0,604	0,607
Корреляция	0,422	0,470	0,164	0,229	0,566	0,376	0,221	-0,230	0,900	0,891	0,889
Скорость роста	0,000	0,002	0,000	0,000	0,002	0,001	0,000	0,000	0,019	0,020	0,020

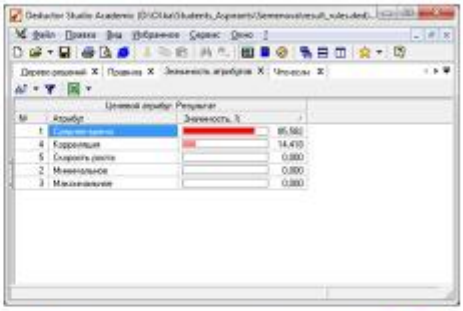
Аналіз результатів

- Правила:

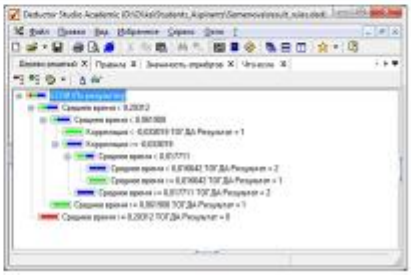
№	Условие	Следствие (Результат)	Поддержка		Достоверность	
			%	Кол-во	%	Кол-во
1	Среднее время < 0,20312 И Среднее время < 0,061908 И Корреляция < -0,033019	1	10,00	1	100,00	1
2	Среднее время < 0,20312 И Среднее время < 0,061908 И Корреляция >= -0,033019 И Среднее время < 0,017711 И Среднее время < 0,016642	2	10,00	1	100,00	1
3	Среднее время < 0,20312 И Среднее время < 0,061908 И Корреляция >= -0,033019 И Среднее время < 0,017711 И Среднее время >= 0,016642	1	10,00	1	100,00	1
4	Среднее время < 0,20312 И Среднее время < 0,061908 И Корреляция >= -0,033019 И Среднее время >= 0,017711	2	40,00	4	100,00	4
5	Среднее время < 0,20312 И Среднее время >= 0,061908	1	10,00	1	100,00	1
6	Среднее время >= 0,20312	0	20,00	2	100,00	2

Аналіз результатів

- значимість атрибутів:



- дерево рішень:



Висновки

- Проаналізовано проблему недостатньої надійності по
- Сформульовано задачу автоматизації оцінки результатів тестування навантаження по
- Побудована система обробки даних і побудови правил оцінки веб додатків

Дякуємо за увагу ☺