КОМП'ЮТЕРНІ СИСТЕМИ І МЕРЕЖІ

Т. 1, № 1, 2019

https://doi.org/

УДК 004.7, 658.012.324

ПРИНЦИПИ ПОБУДОВИ ІНТЕРФЕЙСУ КОРИСТУВАЧА КІБЕРФІЗИЧНОЇ СИСТЕМИ

І. І. Пастернак

Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин

Розглянуто принципи та запропоновано рекомендації щодо розроблення інтерфейсів користувача для кіберфізичної системи. Наведено методи взаємодії кіберфізичної системи з ВЕБ-сервісами, для ефективного їх використання. Реалізовано інтерфейс користувача для кіберфізичної системи у вигляді ВЕБ-сервісу. Подано принципи побудови ВЕБ-сервісів, проведено аналіз наявних архітектур ВЕБ-сервісів. Виділено основні проблемні місця, повязані з проектуванням сервісів на основі RESTта SOAP архітектур. Також розглянуто та описано основні підходи та рекомендації щодо розроблення графічних інтерфейсів користувача для кіберфізичної системи. Подано опис GeoJSON формату даних, який використовують для візуалізації отриманих даних. Наведено методи взаємодії кіберфізичної системи з ВЕБ-сервісами та базою даних. Запропоновано програмне забезпечення, побудоване у вигляді ВЕБ-сервісу на основі архітектури REST. Наведено методи взаємодії кіберфізичної системи з MSSQLServer. Також запропоновано передачу даних, яка відбувається через системи безпровідного зв'язку третього покоління та систему тестів, яка дала змогу сповна перевірити реалізовані функції. Створені перевірки проведено на операційних системах Android від Google, IOS від Apple та Windows компанії Microsoft. Перевірено роботу інтерфейсу користувача для кіберфізичної системи в режимі адміністратора користувача сервісу. Інтерфейс користувача та ВЕБ-сервіс загалом, пройшовши систему тестів, не давали збоїв, не було зафіксофано аномальної поведінки, що свідчить про успішну та правильну реалізацію заявлених функцій.

Ключові слова: веб-сервіс, клієнт, сервер, кіберфізична система.

Вступ

Сучасні досягнення в галузі науки і техніки дають змогу поліпшити зв'язок між обчислювальними і фізичних елементами, що значно збільшує адаптивність, самостійність, ефективність, функціональність, надійність, безпеку і зручність використання КФС. Це дає нові можливості щодо використання КФС у багатьох галузях, зокрема: транспорті, промисловості, сільському господарстві, енергетиці, обороні, охороні здоров'я та інших.

У зв'язку з таким стрімким розвитком КФС, доволі актуальним є питання зручного та інтуїтивно зрозумілого інтерфейсу користувача, що давало б можливість збільшувати швидкість інтеграції таких систем у найрізноманітніші галузі, а реалізація такого інтерфейсу в вигдялі WEB-сервісу забезпечить широкий доступ до ресурсів системи будь-де.

Якщо глянути на міжнародний ринок, то можна побачити, що IT-компанії витрачають значні ресурси, і фінансові, й розумові, на дослідження та розвиток КФС, тому розроблення такого сервісу достатньо важливе. Також, беручи до уваги стрімкий розвиток КФС на ринку України, значні

капіталовкладення в цю галузь та практично цілковиту відсутність напрацювань на цю тему на ринку країн СНД, актуальність цієї роботи цілком виправдана.

Аналіз останніх джерел та публікацій

Кіберфізичні системи поєднують динаміку фізичних процесів із обчислювальними та мережевими рішеннями, забезпечуючи їх спільне й ефективне функціонування в реальному масштабі часу. На відміну від традиційних автономних вбудованих систем, КФС реалізовано у вигляді мережі взаємодіючих елементів із фізичним входом та виходом. Такі системи, як правило, функціонують у режимі зі зворотним зв'язком, за якого фізичні засоби взаємодії впливають на результати обчислень, і навпаки. Сервіси кіберфізичних систем побудовано на основі системи глобального геопозиціонування GPS, часових вимірювань та вимірювань відстаней, що дає змогу визначати координати місцезнаходження рухомого об'єкта та його швидкість, обчислювати відстань й напрямок до пункту прибуття, час прибуття й відхилення від заданого курсу за допомогою приладу трекер. Цей сервіс замість геодезичних знаків і радіомаяків використовує супутники, що випромінюють спеціальні сигнали – поточне місцезнаходження супутників, що знаходяться на високих орбітах [1, 3]. Для визначення відстаней супутники й приймачі генерують складні двійкові кодові послідовності, що називаються псевдовипадковим кодом. Визначення часу розповсюдження сигналу відбувається порівнянням псевдокоду супутника щодо такого самого коду приймача GPS. Кожний супутник має певні власні два псевдовипадкових коди. Щоб розрізняти ці коди та інформаційні повідомлення різних супутників, у приймачі GPS відбувається виклик відповідних функцій. Супутники постійно передають інформацію про своє місцезнаходження. Відстань до них визначають вимірюванням проміжку часу, який потрібний радіосигналу, щоб дійти від супутника до радіоприймача, та множенням його на швидкість розповсюдження електромагнітних хвиль. Точність вимірювання відстаней до супутників забезпечується синхронізацією годинників супутників, у яких використовують атомні еталонні генератори частоти.

Функції кіберфізичних систем:

- збирання великої кількості різнорідної інформації, яка обмежена тільки об'ємом власної пам'яті кіберфізичних систем;
- передача даних кіберфізичних систем до комп'ютера диспетчера, здійснюється через бездротові мережі;
- через використання GPS кіберфізичних систем інформація про переміщення і стан об'єкта доступна диспетчеру в реальному часі (можлива затримка зумовлена способом комунікацій) [4];
- час відгуку у кіберфізичних систем залежить від каналу передачі даних.

Сервіс будують на основі REST архітектури; він надаватиме інформацію про завантаженість доріг у містах. Джерелами збору інформації будуть відкриті GPS-трекери, які входять до складу кіберфізичної системи. Передача інформації здійсюватиметься через безпровідні мережі, оброблятиметься та представлятиметься в інтерфейсі користувача.

Для виконання запланованого завдання вибрано програмне середовище Intellij Idea. Це серія продуктів фірми "JetBrains". Цей продукт дає змогу писати практично будь-якою мовою програмування та має велику кількість додаткових плагінів. Intellij Idea слугуватиме для написання WEB-сервісу та ІК для КФС. Тому, вважаю, створення цього програмного забезпечення необхідне з багатьох причин, серед яких: моніторинг завантаженості доріг у реальному часі, мінімум знань під час роботи з програмним продуктом, зручний і зрозумілий в користуванні інтерфейс користувача. В разі його впровадження та подальшого вдосконалення він принесе велику користь своїм користувачам. Ще одним плюсом буде доступність, адже програмний продукт буде цілком безплатним, що сприятиме його поширенню.

Постановка завдання

Описати основні принципи та рекомендації щодо розроблення інтерфейсів користувача для кіберфізичної системи. Реалізувати інтерфейс користувача як інформаційний сервіс, що взаємодіє з кіберфізичною системою.

Основні результати досліджень

Загальний опис компонентів кіберфізичної системи

Навігаційними пристроями в цій системі є мобільні пристрої користувачів, з вбудованим GPS-передавачем. Комунікація з сервером відчувається через мобільні мережі. Отримавши дані про поточне розташування з супутника, мобільний пристрій передає їх для опрацювання серверу. Загальну структуру кіберфізичної системи та усіх її елементів подано на рис. 1.

Отримавши запит від пристрою користувача, контролер на сервері обробляє отримані дані, зберігає їх в базі даних та повертає інтрефейс користувача, яким є ВЕБ-сервіс, для подальшої взаємодії користувача із сервером [5, 6].

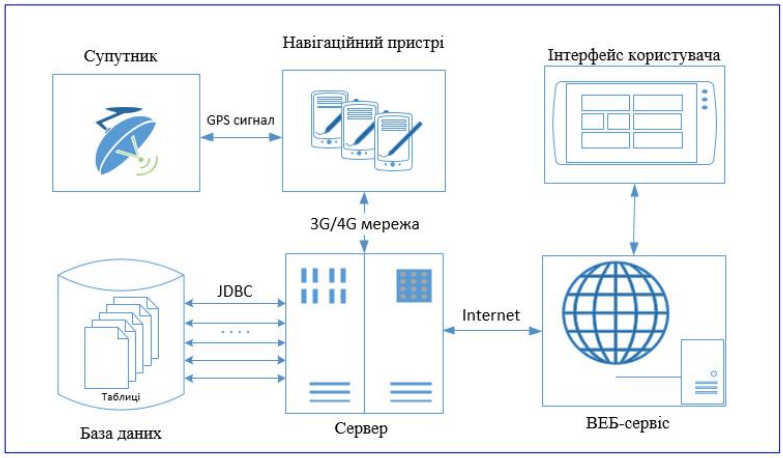

Рис. 1. Загальна структура кіберфізичної системи

Реалізація серверної частини ВЕБ-сервісу

Перед початком реалізації програмного забезпечення необхідно описати його у вигляді діаграм класів. Вони є однією із найпоширеніших форм опису програмних продуктів з огляду на їх проектування, показуючи їх майбутню структуру. На рис. 2 наведено діаграму класів серверної частини системи.

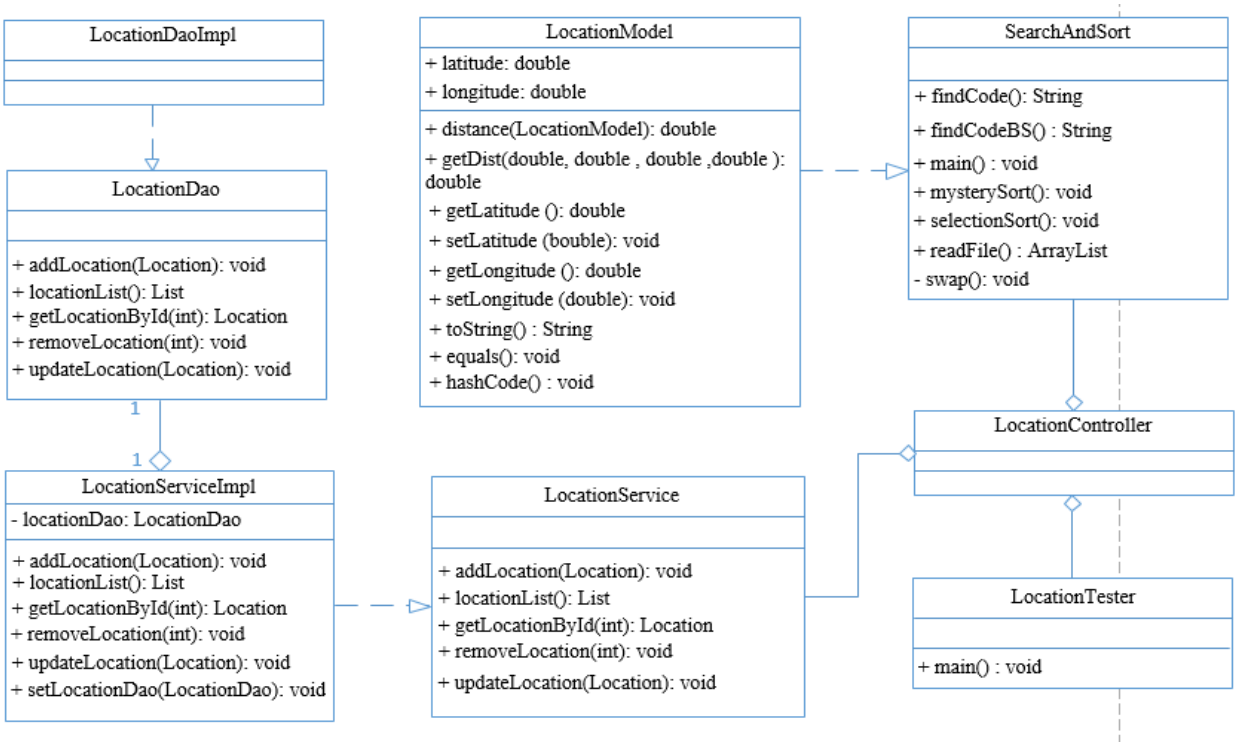

Рис. 2. Діаграма класів серверної частини ВЕБ-сервісу

На ній можна побачити усі класи ВЕБ-сервісу, їх атрибути та відношення з іншими класами. Видно, що в LocationService входять класи LocationServiceImpl, LocationDao, LocationDaoImpl, а сам LocationService входить у клас LocationController, який, крім цього, містить класи SearchAndSort, LocationModel та LocationTester.

LocationModel – клас, що відповідає за модель об'єкта, який обробляється, його поля відповідають полям об'єкта в базі даних, взаємодія відбувається через методи:

- getLatitude (): double – надає можливість отримати широту об'єкта;
- getLongitude (): double – надає можливість отримати довготу об'єкта;
- setLatitude (bouble): void – надає можливість задати широту;
- setLongitude (double): void – надає можливість задати довготу;

Методи – toString() : String, equals(): void та hashCode() : void відповідають за коректність порівняння, переведення та знаходження потрібного об'єкта. Крім цього, методи distance(LocationModel): double та getDist(double, double, double, double): double слугують для отримання відстані між об'єктами на карті [7, 9].

Клас SearchAndSort відповідає за швидке знаходження об'єктів вивантажених із бази даних у ArrayList за допомогою методу readFile() : ArrayList. Методи findCode(): String та findCodeBS() : String відповідають за пошук коду об'єкта.

У методі main() : void викликаються методи – mysterySort(): void та selectionSort(): void, які відповідають за сортування отриманих об'єктів колекції ArrayList. Алгоритм сортування selectionSort().

Для перестановки змінних у сортованому масиві використовують операцію "переприсвоєння" вигляду $A_k = A_i$; $A_i = x$, де k – номер мінімального елемента (одного з двох) на цьому етапі. Словесно алгоритм можна поділити на декілька етапів:

1. Починаємо сортування з першого елемента i = 1.
2. Знайти мінімальний елемент і його номер у масиві $A_1, A_2, ..., A_N$.
3. Поміняти місцями A_i і мінімальний елемент A_k.
4. Перейти до наступного елементу i = i + 1.
5. Якщо розглянуто не всі N – 1 матеріали, то повторити з п. 2.

Інтерфейс LocationDao є прошарком між базою даних та системою. Це дає змогу підвищити рівень абстракції та надалі вносити зміни в код без значної перебудови. Відомі такі методи, як:

- addLocation(Location): void – відповідає за додавання нової локації;
- locationList(): List – дає змогу отримати весь список локацій;
- getLocationById(int): Location – отримати локацію за Id;
- removeLocation(int): void – видалити локацію за Id;
- updateLocation(Location): void – оновити координати локації;

Клас LocationServiceImpl слугує реалізацією для інтерфейсу LocationDao та імплементує перечислені вище методи. Реалізацію цих методів подано на рис. 3.

```
public class LocationDaoImpl implements LocationDao {
   private SessionFactory sessionFactory;
   public void setSessionFactory(SessionFactory sessionFactory) { this.sessionFactory = sessionFactory;}
      Session session = this.sessionFactory.getCurrentSession();
      Location location = session.load(Location.class, new Integer(ID));
   public Location getLocationById(int ID) {
      Session session = this.sessionFactory.getCurrentSession();
      Location location= session.load(Location.class, new
      logger.info("Location успішно завантажився. Інформація: " + Location); return Location;}
```

Рис. 3. Фрагмент коду класу LocationServiceImpl

Інтерфейс LocationService є постачальником правил та методів для сервісу. В ньому описано основні методи, необхідні для комунікації з базою даних. Фрагменти коду наведено на рис. 4.

```
public interface LocationService {
   public void addLocation (Location location);
   public void updateLocation (Location location);
   public void removeLocation (int ID);
   public Location getLocationById(int ID);
   public List< Location> LocationList();
```

Рис. 4. Фрагмент коду класу LocationService

Клас LocationController відповідає за керування ВЕБ-сервісом. За допомогою анотації @RequestMapping з Spring Framework контролер розпізнає, на який URL прийшов запит та виконує необхідну функцію. Під час роботи контролера анотації виконують важливі функції, деякі з них подано нижче:

– org.springframework.beans.factory.annotation.Autowired, яка відповідає за зв'язування на етапі розгортання сервісу чи звертання до контролера, все залежить від налаштування сервера;

– org.springframework.stereotype.Controller – відповідає за ідентифікацію класу як контролера;

– org.springframework.web.bind.annotation.ModelAttribute – дає змогу передати елемент із контролера до інтерфейсу користувача;

– org.springframework.web.bind.annotation.RequestMapping – відповідає за URL, за яким здійснюється звернення до контролера;

Взаємодія з базою даних відбувається за допомогою технології Hibernate для підвищення швидкодії ВЕБ-сервісу, оскільки отримання даних із бази даних часто може значно сповільнювати роботу сервісу [8].

Клієнт, подаючи запит на певний URL, викликає API, яке є прикладним компонентом на сервері. Він, своєю чергою, створює з'єднання з БД та вивантажує потрібну інформацію в ArrayList. Після завантаження даних через Hibermate прикладний компонент обробляє їх та передає для відображення на стороні клієнта. Безпосередньо процес отримання даних повністю покладено на Hibernate. Для його налаштування використовують файл mvc-dispatcher-servlet.xml.

Найважливішими елементами тут є:

– <property name="driverClassName" value="com.mysql.jdbc.Driver"/> – який вказує тип драйвера, що використовується для під'єднання до бази даних;

– <property name="url" value="jdbc:mysql://localhost:3306/projectdb"/> – вказує порт та назву бази даних, до якої виконується під'єднання;

– <property name="username" value="root"/> та <property name="password" value="root"/> вказують ім'я користувача та пароль для під'єднання до сервера бази даних.

Керування додатковими залежностями, завантаженням бібліотек та компонентів фреймворків покладено на Apache Maven.

Для опису всіх необхідних бібліотек використовують файл Pom.xml. Після опису необхідних залежностей та бібліотек Maven автоматично їх завантажує та під'єднує.

Інтерфейс користувача для керування КФС у вигляді ВЕБ-сервісу

Перед реалізацією графічного інтрефейсу користувача кіберфізичної системи необхідно продемонструвати принцип його роботи та комунікації для чіткого розуміння необхідного функціоналу. На рис. 5 наведено функціональну схему ВЕБ-сервісу.

До ВЕБ-сервісу входить база даних, серверний додаток оброблення геоданих, контейнер Hibernate та реалізований обмін даними JSON формату через HTTP протокол. На стороні інтерфейсу користувача кіберфізичної системи реалізовано переведення даних із JSON та подання їх користувачеві. Користувач, своєю чергою, після аналізу отриманого повідомлення, генерує запит на зчитування/запис даних з бази даних чи отримання інформації про поточне розташування, обмеження щодо області пересування та маршруту. Під час розроблення інтерфейсу користувача важливу роль відіграють мовні параметри. Користувачі для поліпшення комунікації з ІК мають можливість обрати мову інтерфейсу.

Також важливим елементом є перемикання виду карти. На вибір користувачеві надають декілька варіантів перегляду карти, серед яких: вигляд із супутника, OSM вигляд та кругова карта. Реалізовано функцію завантаження та відображення даних, що могли бути заздалегідь підготовані. Це дає можливість наперед визначати та зберігати маршрути, виділяти певні ділянки, туристичні області чи задавати іншого роду обмеження. Оскільки часто буває так, що заздалегідь підготовані дані стають неактуальними на момент їх використання, передбачено функцію оновлення цих даних. Це дає змогу в реальному часі змінювати маршрути та задавати нові обмеження. Для підвищення зручності в навігації реалізовано два варіанти пошуку. Перший здійснює пошук за заданим ім’ям та після знаходження переводить користувача на знайдену зону. Інший варіант пошуку реалізований за допомогою координат. У потрібну область вводять координати об’єкта, після чого карта фокусуватиметься на заданих координатах.

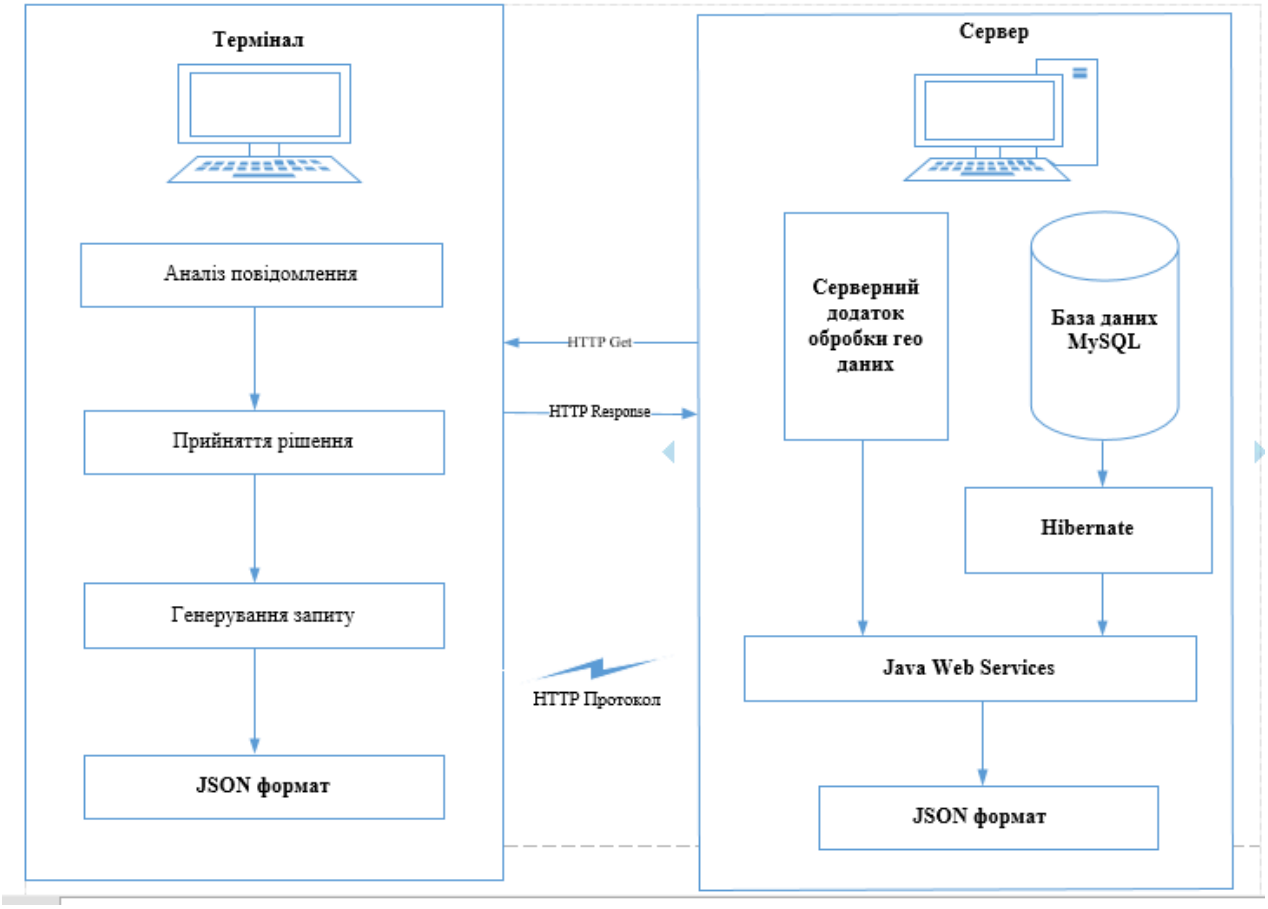

Рис. 5. Функціональна схема ВЕБ-сервісу

Окрім згаданої вище програмної реалізації деяких функцій, передбачена можливість розрахунку відстані. Підрахунок виконують за допомогою широти та довготи двох точок в одному

варіанті та за допомогою полігону точок під час обрахування довжини маршруту в іншому варіанті. Взаємодія користувача з графічним інтерфейсом для КФС. Детальну взаємодію користувача із системою наведено на рис. 6.

Вона здійснюється за допомогою сенсорного дисплею, оскільки більшість сучасних мобільних пристроїв оснащені саме ними, що позбавляє необхідності у використанні додаткових периферичних пристроїв введення/виведення. Звертаючись до сервісу, користувачу виводиться карта та реалізований функціонал. Користувач, отримавши початкові дані, аналізує вміст та приймає рішення щодо подальшої взаємодії з ВЕБ-сервісом. Використавши певну функцію, її обробляють на сервері та за необхідності зберігають у базі даних для подальшого використання. У деяких випадках хотілося б внести до клієнтської частини системи деякі функції для роботи з “локальним кешем” бази даних, тобто з тією її частиною, що інтенсивно використає клієнтська прикладна програма. У сучасній технології це можна зробити тільки, формально створивши на стороні клієнта локальну копію сервера бази даних і розгляду всієї системи як набору взаємодіючих серверів. З іншого боку, іноді хотілося б перенести більшу частину прикладної системи на сторону сервера, якщо різниця в потужності клієнтських робочих станцій і сервера надто велика. Взагалі ж, використовуючи RPC, це зробити неважко. Але потрібно, щоб основне програмне забезпечення сервера дійсно дозволяло це.

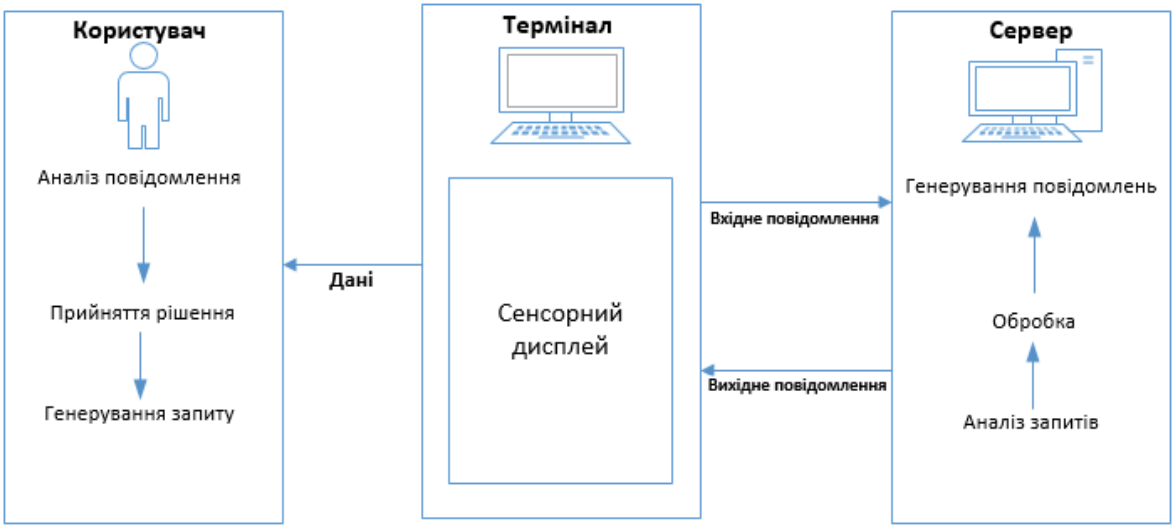

Рис. 6. Схема взаємодії користувача з ГІК у вигляді ВЕБ-сервісу

Тестування інтерфейсу користувача КФС у вигляді ВЕБ-сервісу

Програмна реалізація деяких функцій, описаних у розділі 3, дала змогу отримати кінцевий графічний інтерфейс користувача. Створення системи тестів для перевірки коректності роботи ВЕБ-сервісу та ІК для КФС. Для проведення тестування та перевірки роботи ВЕБ-сервісу необхідно створити систему тестів, яка дасть змогу сповна перевірити зазначені функції. До системи тестів входять такі перевірки: зміни вигляду карт; пошуку за заданим іменем; пошуку за заданими координатами; розрахунку протяжності маршруту; розрахунку відстані між заданими точками; імпорту даних; експорту даних; встановлення полігону; коректності відображення позиції користувача; виходу користувача із зазначеної зони; коректність відображення у мобільних пристроях; коректність роботи функцій сервісу в мобільних пристроях. Реалізовано саме чотири варіанти відображення карти для максимальної зручності користувачів, оскільки кожна з них має свої переваги. Карта OpenStreetMap дає змогу переглядати культурні та релігійні пам’ятки, медичні заклади, театри та бібліотеки. Карта OpenMapSurfer дає змогу оптимально знаходити такі місця, як заправки, готелі, хостели, супермаркети. CycleMap є оптимальною для пошуку закладів харчування, кафе, ресторанів.

Далі тестують можливість пошуку міста, закладу харчування, готелю чи будь-якої іншої локації за назвою.

Після завершення тестування ВЕБ-сервісу та інтерфейсу користувача на стаціонарних пристроях можна приступати до тестування зазначеного функціоналу на мобільних платформах, оскільки не завжди є можливість отримати доступ до стаціонарних комп'ютерів. Як і тестування на стаціонарних пристроях, тестування проводитиметься у двох режимах, а саме в режимі користувача та адміністратора. Мобільними пристроями будуть дві найпопулярніші платформи, а саме IOS фірми Apple та Andriod від Google.

Виконавши тестування описаних функцій, переконавшись, що усі вони коректно працюють на різних платформах, можна порівняти їх з наявними на цей момент, схожими за функціоналом системами. Такими системами є Google Maps та Яндекс Карти. Якщо порівнювати з сервісом від компанії Google, то значною перевагою є те, що передбачені два режими роботи з сервісом, а саме режим адміністратора та користувача. Крім цього, написаний ВЕБ-сервіс має ще один елемент функціоналу, який недоступний в аналогічних сервісах, а саме можливість задання полігону точок, що слугують обмеженням для користувача. Вийшовши із заданої ділянки, користувачу виводиться сповіщення про те, що він покинув задане місце. Якщо зважати на "слабкі" сторони, то мінусом буде масштаб цього ВЕБ-сервісу. Аналоги у вигляді Google Maps та Яндекс Карти є значно потужнішими. Крім цього, в цих сервісах передбачено моніторинг дорожнього трафіку, перегляд знімків у заданих місцях та реалізовано голосову навігацію.

Висновки

У статті описано принципи побудови інтерфейсу користувача дл КФС, а також проаналізовано та оглянуто методи та засоби для побудови ВЕБ-сервісів. Розглянуто види сучасних архітектур, що застосовуються під час побудови ВЕБ-сервісів, а саме REST та SOAP. Визначено "слабкі" місця кожної з них, після чого для кінцевої реалізації вибрано REST архітектуру. Проаналізовано роботу сучасних кіберфізичних систем та елементів, з яких вони складаються. Розглянуто основні підходи до побудови інтерфейсів користувача, після чого виокремлено та поділено на декілька етапів процес розроблення, тестування та оптимізації ІК для КФС.

Проаналізовано програмне середовище, на базі якого реалізовувався ВЕБ-сервіс. Розглянуто та описано додаткові технології, що були необхідні в процесі його реалізації та подальшої коректної роботи, серед яких: "Apache Maven" та "Spring Framework". Вибрано API для реалізації відображення даних, яке, своєю чергою, дало змогу опрацьовувати та відображати GeoJSON та SVG формати даних. Схематично описано роль GPS-модулів мобільних пристроїв користувачів ВЕБ-сервісу та їх взаємодію із сервером. Розглянуто основні функції бази даних у кіберфізичних системах, схематично описано взаємодію сервера бази даних, яким є MSSQL Server, з ВЕБ-сервісом. Також наведено схему взаємодії клієнта з сервісом через інтрефейс користувача кіберфізичної системи.

Схематично описано структуру кіберфізичної системи. Подано ролі та функції усіх елементів КФС. Наведено діаграму класів, за якою проведено подальшу реалізацію ВЕБ-сервісу, описано класи та методи, що відповідають за зберігання, передачу та оброблення даних користувачів. Розроблено та описано алгоритм сортування даних, що використовується у процесі пошуку та впорядкування даних. Також наведено програмну конфігурацію сервера для його ефективної роботи. Під час розроблення графічного інтерфейсу користувача кіберфізичної системи наведено функціональну схему ВЕБ-сервісу, яка дала змогу сповна пояснити взаємодію ІК із бізнес-логікою сервісу. В інтерфейсі користувача реалізовано такі можливості: перемикання мов ГІК, зміна вигляду карт, завантаження та відображення підготованих користувацьких даних, зміна та видалення підготовлених даних, розрахунок відстані маршруту та відстані між вибраними пунктами.

Виконано комплексне тестування ВЕБ-сервісу та інтерфейсу користувача. Запропоновано систему тестів, яка дала змогу сповна перевірити реалізовані функції. До неї увійшли такі перевірки: зміни вигляду карт, пошуку за заданим іменем, пошуку за заданими координатами, розрахунку протяжності маршруту, розрахунку відстані між заданими точками, імпорту даних, експорту даних, встановлення полігону, коректності відображення позиції користувача, виходу користувача із зазначеної зони, на коректність відображення в мобільних пристроях, на коректність роботи функцій сервісу в мобільних пристроях. Створені перевірки виконано на операційних системах Android від Google, IOS від Apple та Windows компанії Microsoft. Здійснено перевірки роботи в режимі адміністратора користувача сервісу. Інтерфейс користувача та ВЕБ-сервіс загалом, пройшовши систему тестів, не давали збоїв, не було зафіксовано аномальної поведінки, що свідчить про успішну та правильну реалізацію заявлених функцій.

Список літератури

1. Липаев В. В. Обеспечение качества программных средств. Методы и стандарты. М.: Синтег, 2001. С. 246.

2. Макгрегор Дж., Сайкс Д. Тестирование объектно-ориентированного программного обеспечения. К.: Диасофт, 2002. С. 432.

3. Тамре Л. Введение в тестирование программного обеспечения. М.: Издательский дом "Вильямс", 2003. С. 368.

4. Татарчук М. І. Корпоративні інформаційні системи: навч. посібник, 2005. С. 245.

5. Мухамедзянов Н. Java. Server applications. Издательство: СОЛОН-Р, 2003. С. 267.

6. Дуглас Камер, Девід Л. Стівенс Сети TCP/IP, т. 3: Разработка приложений типа клиент/сервер, Издательский дом "Вильямс", 2002. С. 592.

7. Фленов М. Є. Web-сервер глазами хакера: Проблемы безопасности Web-серверов; Ошибки в сценариях на PHP, Perl, ASP; SQL-инъекции, 2005. С. 365.

8. Мельник А. О. Кіберфізичні системи: проблеми створення та напрями розвитку // Вісник Національного університету "Львівська політехніка" "Комп'ютерні системи та мережі". 2015, № 692. С. 100–107.

9. Міюшкович Є. Я., Гребеняк А. В., Парамуд Я. С. Телекомунікаційні підсистеми кіберфізичних систем // Вісник Національного університету "Львівська політехніка" "Комп'ютерні системи та мережі". 2016, № 857. С. 65–74.

PRINCIPLES OF CONSTRUCTION A USER INTERFACE FOR CYBERPHYSICAL SYSTEM

I. Pasternak

Lviv Politechnic National University,
Computer Engineering Department

This article discusses a principles and recommendations for developing user interfaces from cyberphysical system. The methods of interaction in cyberphysical system with web-services and database are presented for their effective use. The cyberphysical system user interface is implemented as a web-service. This article discusses the principles and recommendations for developing user interfaces for the cyberphysical system. The methods of interaction the cyberphysical system with web services are presented, for their effective use. The cyberphysical system user interface is implemented as a web-service. This paper examines the principles of building web- services, analyzes the existing architectures of web-services. The main problems related to the design of REST services and SOAP architectures are

highlighted. Also, basic approaches and recommendations for developing graphical user interfaces for cyberphysical system were discussed and described. A description of the GeoJSON data format that is used to visualize the data obtained is given. The methods of interaction of cyberphysical system with web-services and database are presented. The proposed software is built in web-service based on the REST architecture. It is implemented as an information service that interacts with the cyberphysical system. The elements of this system are directly user-friendly, using GPS modules in users' mobile devices. The collection of information is obtained from publicly open docks, with further storage of the data obtained in the database. The methods of interaction of the cyberphysical system with MS SQL Server are given. And, it also suggested data transmission that occurs through third-generation wireless systems. A test system was offered to allow the functions to be fully tested. The checks were created on Google's Android operating systems, iOS from Microsoft's Apple Windows. The user interface for cyberphysical system was verified in service user admin mode. The user interface and web-service as a whole, having passed the system of tests, did not fail, there were no abnormal behavior, which indicates the successful and correct implementation of declared functions.

Key words: web-service, client, server, cyberphysics system.