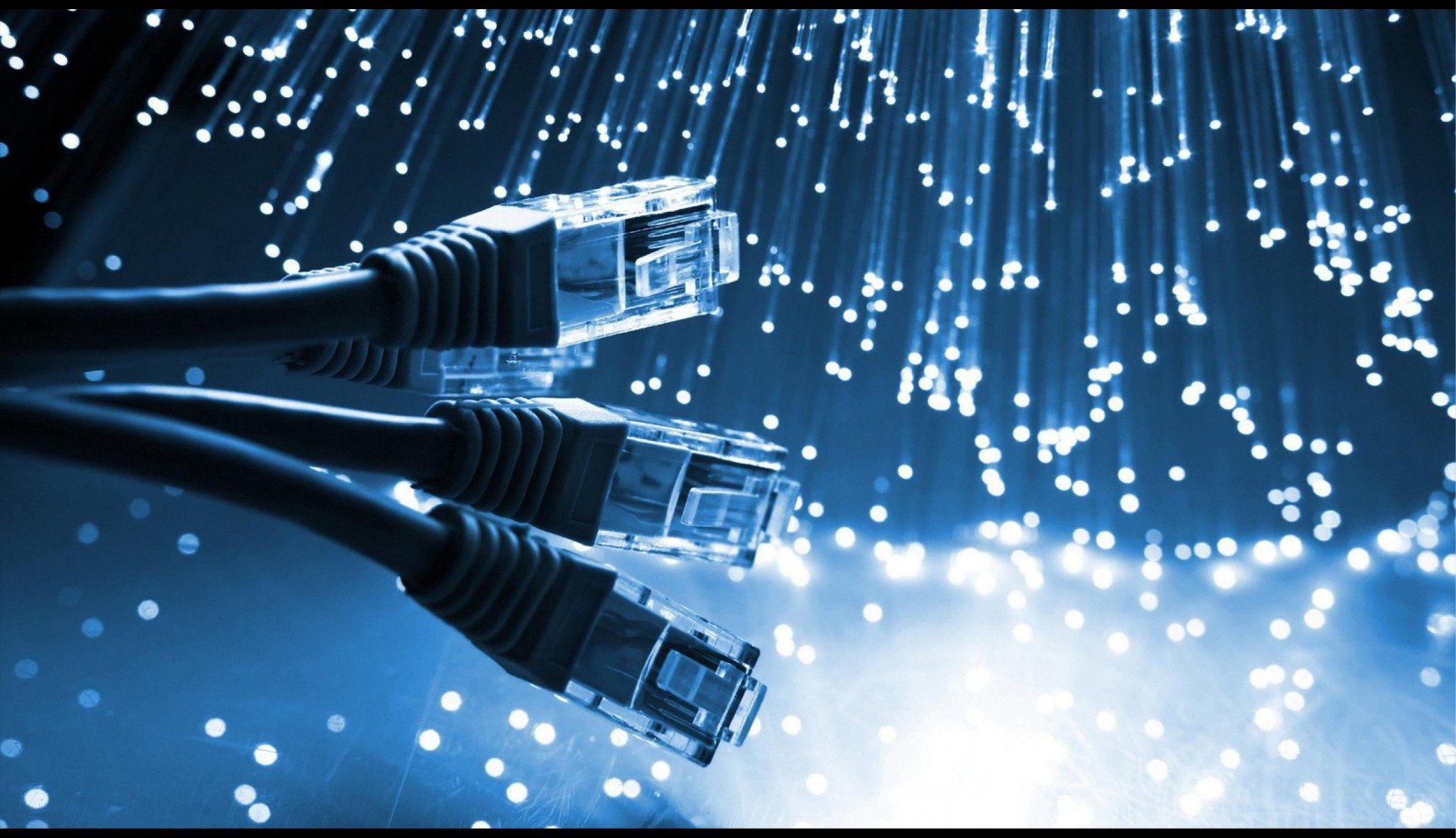
JCSI

Journal of Computer Sciences Institute

Volume 5/2017



Institute of Computer Science
Lublin University of Technology

jcsi.pollub.pl

ISSN: 2544-0764

Redakcja JCSI
e-mail: jcsi@pollub.pl
www: jcsi.pollub.pl
Instytut Informatyki
Wydział Elektrotechniki i Informatyki

Politechnika Lubelska
ul. Nadbystrzycka 36 b
20-618 Lublin

Redaktor naczelny:
Tomasz Zientarski
e-mail: t.zientarski@pollub.pl

Redaktor techniczny:
Beata Pańczyk,
e-mail: b.panczyk@pollub.pl

Recenzenci numeru:
dr Beata Pańczyk
dr inż. Piotr Kopniak
dr Edyta Łukasik
dr Mariusz Dzieńkowski
dr inż. Jacek Kęsik
dr inż. Maria Skublewska-Paszkowska
dr inż. Marek Miłosz
dr inż. Dariusz Gutek
dr inż. Jakub Smołka

Skład komputerowy:
Piotr Misztal
e-mail: p.misztal@pollub.pl

Projekt okładki:
Marta Zbańska

JCSI Editorial
e-mail: jcsi@pollub.pl
www: jcsi.pollub.pl
Institute of Computer Science
Faculty of Electrical Engineering and Computer Science
Lublin University of Technology
ul. Nadbystrzycka 36 b
20-618 Lublin, Poland

Editor in Chief:
Tomasz Zientarski
e-mail: t.zientarski@pollub.pl

Assistant editor:
Beata Pańczyk,
e-mail: b.panczyk@pollub.pl

Reviewers:
Beata Pańczyk
Piotr Kopniak
Edyta Łukasik
Mariusz Dzieńkowski
Jacek Kęsik
Maria Skublewska-Paszkowska
Marek Miłosz
Dariusz Gutek
Jakub Smołka

Computer typesetting:
Piotr Misztal
e-mail: p.misztal@pollub.pl

Cover design:
Marta Zbańska

ISSN 2544-0764

Spis treści

1. **Porównanie wydajności różnych wersji Javy**
   Maciasz Mateusz 142
2. **Analiza jakościowa sygnałów telemetrycznych w oparciu o dane zgromadzone przez dedykowaną aplikację mobilną**
   Damian Wasąg, Małgorzata Plechawska – Wójcik 148
3. **Metoda docinania faz ruchu podczas wiosłowania na ergometrze na podstawie danych trójwymiarowych**
   Kacper Kowalczyk, Maria Skublewska – Paszkowska 155
4. **Analiza porównawcza jakości interfejsów mobilnego dostępu do usług wybranych banków**
   Marcin Kurzyna, Damian Matysiak, Marek Miłosz 159
5. **Analiza wydajnościowa silnika Unity3D w kontekście aplikacji uruchamianych w przeglądarkach internetowych**
   Przemysław Berdak, Małgorzata Plechawska-Wójcik 167
6. **Analiza porównawcza interfejsów mobilnych usług do wyszukiwania połączeń transportu publicznego**
   Filip Skowroński, Marek Miłosz 174
7. **Ocena użyteczności współczesnych narzędzi IT w analizie danych Big Data**
   Rafał Wasiluk, Piotr Muryjas 179
8. **Analiza zastosowania okularów 3D w symulacji przestrzennej na przykładzie wizualizacji architektonicznej**
   Mateusz Karol Smaga, Małgorzata Plechawska-Wójcik 187
9. **Analiza możliwości wykorzystania czujników urządzeń mobilnych pod kontrolą zmodyfikowanych systemów operacyjnych**
   Mateusz Dobrowolski, Michał Dobrowolski, Piotr Kopniak 193
10. **Analiza wpływu refactoringu na jakość kodu – analiza porównawcza dwóch przypadków**
    Mariusz Łukasik, Marek Miłosz 200
11. **Porównanie efektywności szkieletów AngularJS i VueJS**
    Nazar Patrylo, Marek Miłosz 204
12. **Porównanie wydajności platformy Xamarin oraz aplikacji natywnych na przykładzie systemu operacyjnego Android**
    Ihor Bodia, Małgorzata Plechawska-Wójcik 208
13. **Analiza możliwości wykorzystania bibliotek Qt w wybranych zagadnieniach komunikacji międzyprocesowej**
    Łukasz Janusz Górniak, Maciej Pańczyk 213
14. **Ocena stopnia wykorzystania e-usług w administracji samorządowej w województwie lubelskim**
    Daniel Ziętek, Elżbieta Miłosz 220

Contents

1. **PERFORMANCE COMPARISON OF DIFFERENT JAVA VERSIONS**
   MACIASZ MATEUSZ....142
2. **QUALITATIVE ANALYSIS OF TELEMETRY SIGNALS BASED ON DATA GATHERED BY DEDICATED MOBILE APPLICATION**
   DAMIAN WASĄG, MAŁGORZATA PLECHAWSKA – WÓJCIK....148
3. **METHOD OF CUTTING PHASES OF MOTION WHILE ROWING ON THE ERGOMETER BASED ON THREE-DIMENSIONAL MOTION**
   KACPER KOWALCZYK, MARIA SKUBLEWSKA – PASZKOWSKA....155
4. **A COMPARATIVE ANALYSIS OF INTERFACE QUALITY OF MOBILE ACCESS TO THE SERVICES OF SELECTED BANKS**
   MARCIN KURZYNA, DAMIAN MATYSIAK, MAREK MIŁOSZ....159
5. **PERFORMANCE ANALYSIS OF UNITY3D ENGINE IN THE CONTEXT OF APPLICATIONS RUN IN WEB BROWSERS**
   PRZEMYSŁAW BERDAK, MAŁGORZATA PLECHAWSKA-WÓJCIK....167
6. **COMPARATIVE ANALYSIS OF MOBILE INTERFACES SERVICES TO SEARCH PUBLIC TRANSPORT CONNECTIONS**
   FILIP SKOWROŃSKI, MAREK MIŁOSZ....174
7. **THE ASSESSMENT OF USEFULNESS MODERN IT TOOLS OF DATA ANALYSIS BIG DATA**
   RAFAŁ WASILUK, PIOTR MURYJAS....179
8. **ANALYSIS OF THE 3D GLASSES APPLICATION IN A SPATIAL SIMULATION ON THE BASIS OF THE ARCHITECTURAL VISUALIZATION**
   MATEUSZ KAROL SMAGA, MAŁGORZATA PLECHAWSKA-WÓJCIK....187
9. **ANALYSIS OF THE USE OF SENSORS IN MOBILE DEVICES WITH MODIFIED OPERATING SYSTEMS**
   MATEUSZ DOBROWOLSKI, MICHAŁ DOBROWOLSKI, PIOTR KOPNIAK....193
10. **ANALYSIS OF THE IMPACT OF REFACTORING ON CODE QUALITY – COMPARATIVE ANALYSIS OF TWO CASES**
    MARIUSZ ŁUKASIK, MAREK MIŁOSZ....200
11. **COMPARISON OF ANGULARJS AND VUEJS FRAMEWORKS EFFICIENCY**
    NAZAR PATRYLO, MAREK MIŁOSZ....204
12. **PERFORMANCE COMPARISON OF THE XAMARIN PLATFORM AND NATIVE APPLICATIONS FOR ANDROID OPERATING SYSTEM**
    IHOR BODIA, MAŁGORZATA PLECHAWSKA-WÓJCIK....208
13. **ANALYSIS OF QT LIBRARIES USAGE IN SELECTED INTERPROCESS COMMUNICATION APLICATIONS**
    ŁUKASZ JANUSZ GÓRNIAK, MACIEJ PAŃCZYK....213
14. **EVALUATION OF THE UTILIZATION RATE OF E-SERVICES PROVIDED BY THE LOCAL ADMINISTRATION IN LUBELSKIE VOIVODESHIP**
    DANIEL ZIĘTEK, ELŻBIETA MIŁOSZ....220

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 142-147
Wysłane: 2017-06-18
Przyjęte: 2017-06-21

Porównanie wydajności różnych wersji Javy

Maciasz Mateusz*
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W niniejszym artykule opisano wyniki badań wydajności dwóch wersji Javy - 1.7 i 1.8. Do tego celu zostały stworzone trzy aplikacje. Pierwsza z nich jest odpowiedzialna za przygotowanie danych do testów. Dwie kolejne implementują testy wydajnościowe w zależności od wersji wirtualnej maszyny Javy. Stworzone metody w aplikacjach testowych, miały za zadanie przeanalizować wydajność operacji na kolekcjach oraz szybkości zmiany wartości zmiennych przez dwa wątki.

Słowa kluczowe: Java; Wydajność; Programowanie; Optymalizacja

*Autor do korespondencji.

Adres/adresy e-mail: mateuszmaciasz92@gmail.com

Performance comparison of different Java versions

Maciasz Mateusz*
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This article describes the performance results of two versions of Java - 1.7 and 1.8. Three applications were created for this purpose. The first one is responsible for preparing the test data. Two more implement performance tests depending on the virtual machine version of Java. The methods in the test applications were designed to analyze the performance of collections and the rate of change of the variables by two threads.

Keywords: Java; Performance; Programming; Optimization

*Corresponding author.

E-mail address/addresses: mateuszmaciasz92@gmail.com

1. Wstęp

W dzisiejszych czasach niewiele osób zwraca uwagę na optymalizację, ze względu na szybkość rozwoju technologii i coraz większe zapasy pamięci oraz bardziej wydajne procesory, co sprawia że problem wydajności staje się coraz bardziej realny i namacalny.

Dynamiczny rozwój *Javy* w ostatnich latach zmienił podejście do sposobu programowania w tym języku. Od marca 2014 [1] dostępna jest wersja JDK 1.8 wprowadzająca całkowicie inną koncepcję programowania w porównaniu do dotychczasowej. Język ten stał się bardziej funkcyjny niż dotychczas. Został także rozszerzony o możliwość prostszego programowania równoległego. Nowe API [1] wprowadza dużo uproszczeń i zmian oraz kilka optymalizacji do już istniejących metod. Jedną z podstawowych nowości są wyrażenia *lambda* dostępne dotychczas w innych językach programowania. Zostały także dodane strumienie - rozumiane jako widoki kolekcji. Dzięki strumieniom istnieje możliwość korzystania z operacji typu [2] *map-reduce,* służących do przepakowywania i obróbki danych.

Ze względu na fakt wprowadzenia tak dużej liczby zmian w stosunku do poprzedniej wersji *Javy,* nasuwa się pytanie o wydajność. Jak szybko nowe metody radzą sobie z danymi w stosunku do dawniej używanych implementacji.

2. Przegląd literatury

W ostatniej dekadzie przemysł skupił się na produkcji procesorów wielordzeniowych w celu zwiększenia wydajności [2]. Wcześniej próbowano zwiększać szybkość taktowania procesorów, co skutkowało malejącymi zyskami z powodu czynników takich jak: zużycie energii, które wzrasta wykładniczo wraz ze wzrostem częstotliwości. Taki obrót spraw spowodował powstanie równoległych sposobów przetwarzania danych. JDK w wersji 8.0 zostało wydane w marcu 2014. Api *Javy 1.8* wprowadza przejrzyste i bezpieczne wątkowo metody dostępne dla użytkowników nie posiadających zaawansowanej wiedzy w zakresie programowania równoległego [2]. Metody te bazują na stylu programowania funkcyjnego poprzez wykorzystanie operacji typu *map-reduce* [2]. Operacje tego typu istniały już w innych językach programowania, a zaczęły być powszechnie stosowane w miarę wzrostu zapotrzebowania przetwarzania równoległego. Wielu ekspertów twierdzi że wprowadzenie nowych funkcji programowania t.j. wyrażeń *lambda*, operacji *map-reduce* oraz *strumieni,* jest pod wieloma względami, największą zmianą w języku *Java* w ciągu ostatnich kilku lat.

W [3] autorzy zwracają uwagę na olbrzymią rolę jaką odegrał *JIT (just-in-time) compiler* w języku programowania jakim jest *Java.* Dzięki takiemu narzędziu język ten zyskał bardzo na wydajności. W momencie ukazania się *Javy 1.8* autorzy postanowili stworzyć własny kompilator *JIT*

wykorzystujący procesor graficzny (GPU) [3]. Dzięki nowemu API dostarczonemu w *Javie 1.8*, istnieje więcej możliwości programowania równoległego niż w poprzednich wersjach.

W [4] został poruszony temat wydajności *Javy* w odniesieniu do operacji liczbowych. *Java* jest językiem interpretowanym co może powodować powolne operacje na liczbach [4]. Maszyna wirtualna *HotSpot* implementuje nowy i ulepszony *garbage collector* oraz posiada ulepszoną synchronizację. Rozwiązuje także niektóre problemy wydajnościowe za pomocą adaptacyjnej technologii optymalizacji [5]. Większość aplikacji wykonuje tylko niewielką część kodu źródłowego. *HotSpot* posiada interpreter, z którego pomocą aplikacja zostaje uruchomiona. Następnie z jego pomocą kod jest analizowany w kierunku wyszukiwania "gorących punktów". Następnie miejsca te są kompilowane do kodu natywnego. Ponadto *HotSpot* zawiera wszystkie klasyczne optymalizacje, takie jak: eliminacja martwego kodu, wspólna interpretacja, rozwijanie pętli.

W [6] można znaleźć informacje o czynnikach przeszkadzających *Javie* osiągnąć wysoką wydajność. Jednym z nich jest dynamiczne zarządzanie pamięcią. Wszystkie obiekty *Javy* są przydzielone do puli i podlegają grupowaniu, przenoszeniu i odśmiecaniu. Tablice wielowymiarowe są w rzeczywistości wektorami obiektów odwołujących się do obiektów tablicowych o niższym poziomie [6]. Nie ma zatem możliwości zagwarantowania że obiekty te są przyporządkowane do pamięci o takim samym rozmiarze. Tak więc wszelkie algorytmy i transformacje programowo rzadko zależne od wiersza wydają się być przenośne, trudno jest zagwarantować, że takie przekształcenia są w rzeczywistości możliwe i skuteczne. Ponadto mimo gwarancji założenia pewnego układu pamięci, może on zostać unieważniony przez przydziały i odśmiecanie pamięci.

Oryginalnym modelem *Javy* jest interpretacja kodu na kod bajtowy [6]. Kompilatory *JIT* zwykle przekładają kod stosu JVM na kod rejestru RISC dla zoptymalizowania realizacji operacji. Jednak w przeciwieństwie do C, nie jest jasne czy oryginalna optymalizacja na poziomie kodu źródłowego wykonana przez użytkownika czy tłumaczenie wysokiej jakości kodu źródłowego do kodu bajtowego jest skuteczne. Innymi słowy, nie jest jasne, czy jakikolwiek rodzaj optymalizacji na poziomie użytkownika lub na wysokim poziomie spowoduje zwiększenie wydajności, czy utratę wydajności [5].

3. Cel i zakres badań

Celem badań jest porównanie wydajności dwóch wersji *Javy* - 1.7 i 1.8. Ten język programowania w ostatnim czasie rozwija się bardzo dynamicznie. Zmiany pomiędzy wersjami są znaczne i wprowadzają wiele udogodnień lub też tworzą całkowicie nowe możliwości. Pomiędzy w/w wersjami widać różnice, które zmieniają sposób programowania w tym języku, jak również umożliwiają wybór sposobu rozwiązywania problemów.

Zakres pracy obejmuje następujące zagadnienia:

- Analizę wydajności operacji na kolekcjach.
- Porównanie szybkości działania kolekcji z Javy 1.7 i Stream Api z Javy 1.8. [1]
- Wykazanie różnic pomiędzy wersjami Javy.
- Analizę możliwości nowego Api.

Do badania szybkości wykonywania operacji i metod użyto biblioteki *JMH - Java Microbenchmark Harness* [6].

4. Obiekt badań

W celu zbadania wydajności dwóch wersji Javy, zostały stworzone dwie aplikacje zawierające testy porównawcze oparte na bibliotece JMH. *Java Microbenchmark Harness* [6]. Jest to narzędzie, które pomaga poprawnie tworzyć metody do analizy porównawczej. Narzędzie to zostało stworzone przez tą samą grupę osób, która stworzyła wirtualną maszynę Javy.

Napisanie dobrej metody do analizy porównawczej nie jest rzeczą łatwą. Istnieje wiele sposobów optymalizacji, które stosuje wirtualna maszyna lub warstwa sprzętowa, dla komponentów wyizolowanych do celów analizy. Takie optymalizacje mogą nie mieć miejsca w podobnych metodach wykorzystywanych jako część większej aplikacji. Źle stworzona analiza metody może spowodować że jej wynik będzie świadczył o znacznie wyższej wydajności niż jest ona w rzeczywistości. Poprawne pisanie metod do analizy wydajnościowej JVM pociąga za sobą konieczność uniemożliwienia wirtualnej maszynie oraz zasobom sprzętowym optymalizacji, która nie mogłaby być zastosowana w systemie produkcyjnym [7]. Podczas tworzenia metod analizy należy pamiętać że kompilując kod *Javy* do postaci kodu bajtowego, jest on optymalizowany. Metody które przetwarzają dane lecz nie zwracają żadnych wyników wykonanych operacji są automatycznie traktowane przez kompilator jako martwy kod, co sprawia że w rezultacie metody te wydają się szybkie i optymalne. W celu uniknięcia takich błędów należy zadbać o to aby metody analizy zwracały dane. Istnieje również inny sposób wymyślony przez twórców JMH. Na potrzeby kompilacji została utworzona klasa o nazwie [6] *Blackhole*. Jej zadaniem jest konsumowanie powstałych, w procesie analizy, danych. W ten sposób można łatwo uniknąć martwego kodu.

5. Metody testujące wydajność JVM

Aplikacja generująca dane zwraca kolekcję encji o strukturze przedstawionej na listingu 1.

Przykład 1. Testowa encja danych.

```
public class Person {

  private Long personId;
  private String firstName;
  private String lastName;
  private Integer age;
  private Sex sex;
  private Date birthDate;
  private Integer growth;
  private List<Person> children;
  private Person partner;

}
```

Metoda generująca zestaw danych tworzy kolekcję obiektów wypełnioną przypadkowo wygenerowanymi wartościami. Stworzone w ten sposób kolekcje posiadają strukturę umożliwiającą operowanie takimi metodami jak np: przepakowywanie czy wyszukiwanie w różnych warstwach danych.

Analiza wydajności została przeprowadzona na kolekcji zawierającej 100 000 rekordów. Wygenerowanie takiej ilości danych trwa około 30 sekund. JVM potrzebuje czasu aby zacząć działać optymalnie. W związku z tym przeprowadzone testy zostały poprzedzone 5 iteracjami rozgrzewającymi wirtualną maszynę *Javy*, a następnie wykonywano 20 iteracji testowych. Uzyskany w ten sposób rezultat został przedstawiony w mikrosekundach / operację jako średni czas wykonania metody.

5.1. Przepakowanie kolekcji do listy par.

Test przepakowania listy zawierającej encje typu Person został zrealizowany w sposób przedstawiony na listingu 2. Nowo powstała kolekcja składała się z par rekordów.

Przykład 2. Przepakowanie listy do kolekcji par - *Java 1.8.*

```
@BenchmarkMode({Mode.AverageTime})
@OutputTimeUnit(TimeUnit.MICROSECONDS)
@State(Scope.Thread)
public class CollectionsOperations {
  @Benchmark
  public List<Pair<Person, Person>> pairPersons() {
     return Data.getPersons().stream()
           .reduce(new ArrayList<>(), (sum, element) -> {
              sum.add(
               Pair.newPair(element, element.getPartner())
              );
              return sum;
           }, (sum1, sum2) -> {
              sum1.addAll(sum2);
              return sum1;
           });
  }
}
```

Zwrócone przez generator dane traktowane są jako widok. W następnym kroku z pomocą metody *reduce* tworzona jest kolekcja o zadanej strukturze. Kolejne parametry definiują sposób w jaki ma zostać przetworzona lista aby otrzymać spodziewany efekt. Podobnie zostało to zrealizowane w *Javie 1.7.* Listing 3 prezentuje implementację testu.

Przykład 3. Przepakowanie listy do kolekcji par - *Java 1.7.*

```
@BenchmarkMode({Mode.AverageTime})
@OutputTimeUnit(TimeUnit.MICROSECONDS)
@State(Scope.Thread)
public class CollectionsOperations {

  @Benchmark
  public List<Pair<Person, Person>> pairPersons() {
     final List<Pair<Person, Person>> personPairs =
                                        new ArrayList<>();
     for (final Person person : Data.getPersons()) {
        personPairs.add(Pair.newPair(person,
person.getPartner()));
     }
     return personPairs;
  }
}
```

5.2. Zmiana wartości logicznej.

Zmiana wartości zmiennej logicznej na przeciwną została zaimplementowana jako jeden z testów analizy wydajnościowej. Sposób w jaki został stworzony ten test został przedstawiony na listingu 4.

Przykład 4. Zmiana wartości logicznej - *Java 1.8 i 1.7.*

```
@State(Scope.Group)
@BenchmarkMode({Mode.AverageTime})
@OutputTimeUnit(TimeUnit.NANOSECONDS)
@Threads(2)
public class PingPong {
  private static final String pingPong = "pingPong";
  private final AtomicBoolean boolFlag = new
AtomicBoolean();

  private static void doNothing() {
  }

  @Benchmark
  @Group(pingPong)
  public void ping(Control cnt) {
     while (!cnt.stopMeasurement &&
           !boolFlag.compareAndSet(false, true)) {
        doNothing();
     }
  }
  @Benchmark
  @Group(pingPong)
  public void pong(Control cnt) {
     while (!cnt.stopMeasurement &&
           !boolFlag.compareAndSet(true, false)) {
        doNothing();
     }
  }
}
```

W tym teście dwa wątki zmieniają wartość logiczną zmiennej gdy spełniony jest prosty warunek. Do poprawnego działania testu niezbędne jest skorzystanie z bezpiecznej wątkowo implementacji *AtomicBoolean.* Dodatkowo dwie metody posiadają tą samą grupę dzięki czemu jedna iteracja wywołuje obie metody jeden raz. Jedynym parametrem obu tych funkcji jest klasa *Control*, która przechowuje 2 flagi - *startMeasurement* oraz *stopMeasurement* [6]. Dzięki niej można zatrzymać wykonywanie testu unikając nieskończonej pętli. Powyższy test sprawdza szybkość zmiany wartości logicznej pomiędzy dwoma wątkami.

5.3. Zmiana wartości liczbowej.

Następny test sprawdza szybkość zmiany wartości liczbowej z 0 na 1 i odwrotnie. Test wygląda analogicznie jak poprzedni, który sprawdzał szybkość zmiany wartości logicznej. Kod odpowiadający za analizę przykładu znajduje się na listingu 5.

Przykład 5. Zmiana wartości liczbowej - *Java 1.8 i 1.7.*

```
@State(Scope.Group)
@BenchmarkMode({Mode.AverageTime})
@OutputTimeUnit(TimeUnit.NANOSECONDS)
@Threads(2)
public class PingPong {
```

```
  private static final String intPingPong = "intPingPong";

  private final AtomicInteger intFlag = new
AtomicInteger();

  private static void doNothing() {

  }

  @Benchmark
  @Group(intPingPong)
  public void intPing(Control cnt) {
    while (!cnt.stopMeasurement &&
!intFlag.compareAndSet(0, 1)) {
      doNothing();
    }
  }

  @Benchmark
  @Group(intPingPong)
  public void intPong(Control cnt) {
    while (!cnt.stopMeasurement &&
!intFlag.compareAndSet(1, 0)) {
      doNothing();
    }
  }
}
```

Podobnie jak poprzednio, dwa wątki konkurujące ze sobą zmieniają wartość liczbową, co zostało przedstawione na powyższym przykładzie. Jednokrotne wywołanie testu powoduje uruchomienie pięciu iteracji rozgrzewających wirtualną maszynę *Javy,* a następnie dziesięciu iteracji testowych, z których każda uruchamia jeden raz metodę *intPing* i *intPong*. Wynikiem testu jest średni czas wykonania operacji mierzony w nanosekundach. Test ten ma na celu zmierzenie czasu potrzebnego na zmianę wartości zmiennej liczbowej.

6. Wyniki badań

Badania zostały przeprowadzone z wykorzystaniem dwóch wersji *Javy*. Pierwszą z nich była wersja JDK 1.7.0_80, VM 24.80-b11, a drugą JDK 1.8.0_131, VM 25.131-b11. Ponadto testy zostały przeprowadzone na dwóch różnych stacjach roboczych - komputerze stacjonarnym oraz laptopie. Komputer stacjonarny posiadał system operacyjny *Windows 7 Home Premium 64 bit*, procesor *Intel® Core™ i5-4460 CPU @3.20 GHz 3.20 GHz,* 16,0 GB pamięci RAM, a laptop - system operacyjny *Ubuntu 16.04 LTS 64 bit,* procesor *Intel® Core™ i7-2860QM CPU @2.50 GHz x 8, 32,0 GB pamięci RAM.*

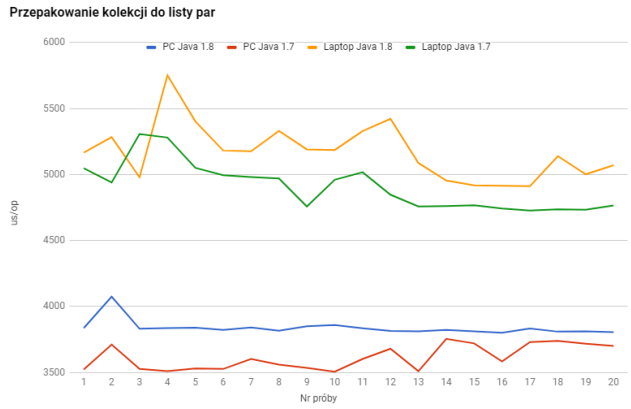
Każdy test składał się z 5 iteracji rozgrzewających wirtualną maszynę *Javy*, 20 iteracji pomiarowych i został wykonany jeden raz. Testy były wykonywane z wykorzystaniem jednego wątku, z wyjątkiem tych metod, które potrzebowały więcej niż jednego wątku, co zostało jednoznacznie zaznaczone przy opisie metody.

6.1. Przepakowanie kolekcji do listy par.

Operacja przepakowania kolekcji zawierającej encje *Person* do listy par dwóch encji *Person* w sposób pokazany na listingu 2, dostarczyła wyniki, które zostały zaprezentowane w tabeli 1. Rys. 1 obrazuje dane z tabeli. Na tej podstawie można wywnioskować że najszybciej test został wykonany na maszynie wirtualnej *Javy 1.7* na komputerze stacjonarnym. Najwięcej czasu potrzebowała *Java 1.8* na laptopie. Można także zaobserwować mniejszą wydajność laptopa w odniesieniu do komputera stacjonarnego. Odchylenia standardowe wyników testu wahają się od 80,00 do 200,00.

Tabela 1. Wyniki operacji przepakowania kolekcji do listy par.

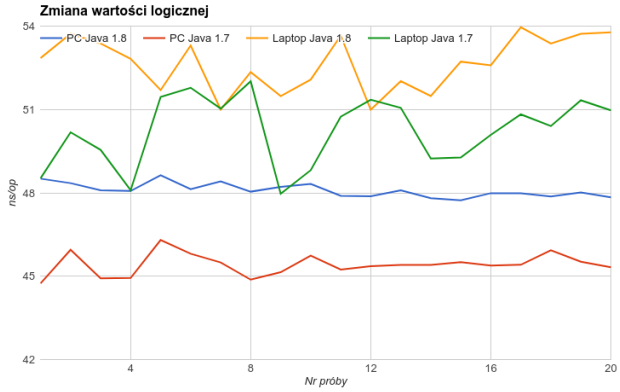
Nr próby	PC Java 1.8	PC Java 1.7	Laptop Java 1.8	Laptop Java 1.7
1	3836,142 us/op	3522,768 us/op	5165,445 us/op	5047,974 us/op
2	4073,756 us/op	3711,410 us/op	5282,647 us/op	4939,661 us/op
3	3830,732 us/op	3526,263 us/op	4979,250 us/op	5307,038 us/op
4	3835,686 us/op	3510,052 us/op	5751,930 us/op	5280,036 us/op
5	3838,824 us/op	3530,200 us/op	5403,282 us/op	5050,131 us/op
6	3822,842 us/op	3526,287 us/op	5181,453 us/op	4994,288 us/op
7	3841,086 us/op	3601,329 us/op	5176,269 us/op	4980,575 us/op
8	3816,005 us/op	3559,296 us/op	5330,672 us/op	4970,284 us/op
9	3849,990 us/op	3534,492 us/op	5190,490 us/op	4757,268 us/op
10	3858,492 us/op	3505,599 us/op	5185,868 us/op	4960,080 us/op
11	3835,131 us/op	3602,111 us/op	5329,542 us/op	5016,783 us/op
12	3814,478 us/op	3679,456 us/op	5421,572 us/op	4846,614 us/op
13	3812,144 us/op	3509,353 us/op	5087,040 us/op	4757,908 us/op
14	3822,110 us/op	3754,258 us/op	4954,094 us/op	4760,562 us/op
15	3811,074 us/op	3719,913 us/op	4918,205 us/op	4766,285 us/op
16	3801,343 us/op	3582,895 us/op	4915,186 us/op	4741,753 us/op
17	3832,891 us/op	3730,054 us/op	4912,205 us/op	4727,307 us/op
18	3809,141 us/op	3738,729 us/op	5138,977 us/op	4736,572 us/op
19	3810,941 us/op	3717,606 us/op	5001,970 us/op	4732,542 us/op
20	3804,932 us/op	3699,994 us/op	5070,013 us/op	4765,845 us/op
średnia	3837,887 us/op	3613,103 us/op	5169,805 us/op	4906,975 us/op



Rys 1: Wyniki operacji przepakowania kolekcji do listy par.

6.2. Zmiana wartości logicznej.

Wyniki testu zmieniającego wartość logiczną zmiennej przez dwa wątki zostały pokazane na rys. 2. Należy zwrócić uwagę na jednostkę w jakiej zostały przedstawione wyniki. Dotychczas były to µs/op, natomiast w tym przypadku są to ns/op, ze względu na bardzo krótki czas wykonania jednej operacji. Na wykresie widać wyraźnie niewielką przewagę

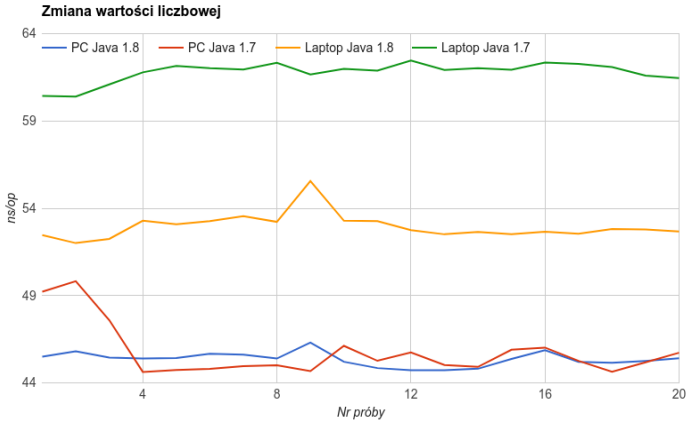


Rys 2: Zmiana wartości logicznej

Javy 1.7 nad *Javą 1.8,* oraz przewagę komputera stacjonarnego nad laptopem. Wartości odchylenia standardowego dla poniższych wyników to ok. 0,3 dla PC i ok 2,0 dla laptopa. Świadczy to o mniejszych wahaniach wyników na komputerze stacjonarnym.

6.3. Zmiana wartości liczbowej.

Rezultaty uzyskane podczas wykonywania testu zmieniającego wartość liczbową zmiennej z 0 na 1 przez dwa wątki, zostały zaprezentowane na rys.3. Na podstawie wykresu można zauważyć, że najwięcej czasu potrzebowała *Java 1.7* na laptopie. *Java 1.7 i 1.8* na PC osiągnęły podobne rezultaty. Wszystkie odchylenia standardowe dla wyników przeprowadzonych testów oscylowały wokół wartości 1,00, co świadczy o niewielkim odchyleniu wyników od średniej.



Rys 3: Zmiana wartości liczbowej

7. Wnioski

Celem testów było zbadanie wydajności wirtualnej maszyny Javy w zależności od użytej wersji oraz platformy uruchomieniowej. Do testów zostały użyte dwie wersje Javy - 1.7 i 1.8, a testy były uruchamiane na dwóch różnych komputerach - komputerze stacjonarnym i laptopie. Na potrzeby testów zostały stworzone trzy niewielkie projekty. Pierwszy projekt został stworzony z myślą o dostarczeniu danych do testów. Kolejne dwa projekty zawierały metody testowe oparte o *Java Microbenchmark Harness* - framework służący do próbkowania i badania wydajności metod i klas *Javy* [7], oraz różne wersje *JDK*.

Na podstawie otrzymanych rezultatów można zaobserwować wyższą wydajności w *Javie 1.8* podczas wykonywania metod służących do przepakowania danych z jednych struktur do innych. Jednocześnie zauważono że wydajność ta jest mniejsza niż w Javie 1.7, w testach zmieniających wartość zmiennych logicznych. Otrzymane wyniki testów posiadały zazwyczaj niewielkie wartości odchylenia standardowego co świadczy o niewielkiej amplitudzie wartości. Wszystkie testy zostały wykonane jeden raz. Każdy test posiadał także podsumowanie zawierające najmniejszy, średni i największy czas wykonania oraz uzyskane odchylenie standardowe. W większości testów, do ich wykonania, użyty został jeden wątek a wyniki zostały przedstawione w jednostce µs/op. Testy zmiany wartości zmiennych liczbowych i logicznych, wykorzystywały dwa lub więcej wątków oraz ich wyniki przedstawione zostały w jednostce ns/op ze względu na bardzo krótki czas wykonania.

Utworzone projekty z testami wydajnościowymi można dowolnie rozwijać implementując kolejne testy sprawdzające wydajność wirtualnej maszyny *Javy*. Na potrzeby artykułu powstały szkielety aplikacji, gotowe do dalszego rozwoju. Tworzenie testów wydaje się dosyć łatwe pod warunkiem zachowania zasad poprawności ich implementowania. Należy pamiętać m.in. aby metody testowe były jak najbardziej zbliżone do produkcyjnie wykorzystywanych metod. Kryteria którymi należy się kierować to przede wszystkim konieczność zwracania danych [6], lub korzystanie z możliwości jakie dostarcza *JMH,* czyli *blackhole,* które konsumują wyniki.

W przypadku niezastosowania powyższych zasad, powstały kod jest traktowany jako *martwy* [6], co powoduje błędne wyniki testów, ze względu na optymalizację zastosowaną przez *JVM*. Duży wpływ na otrzymane rezultaty ma też środowisko na jakim testy zostaną uruchomione. Mocniejsze maszyny osiągają znacznie lepsze rezultaty niż komputery o słabszej wydajności i specyfikacji sprzętowej.

Istnieje możliwość poprawienia wydajności wirtualnej maszyny poprzez dostosowanie opcji uruchomieniowych. Dzięki temu widać możliwość rozwinięcia pracy o próby konfiguracji *JVM* tak, aby osiągnąć lepsze rezultaty powstałych testów. Takie opcje można zmieniać w trakcie wykonywania testów, co pozwala na dynamiczne dostosowywanie wirtualnej maszyny, w sposób pozwalający na optymalizację wydajności. Przykładowo można stworzyć metodę opartą na algorytmie genetycznym, wybierającą najlepsze ustawienia i modyfikującą je w niewielki sposób, w poszukiwaniu wydajniejszych ustawień.

Literatura

[1] Java Platform, Standard Edition (Java SE) 8, 1993, 2016, Oracle and/or its affiliates, http://docs.oracle.com/javase/8/docs/api/

[2] Andres R. Masegosa, Ana M. Martinez, Hanen Borchani, Probabilistic Graphical Models on Multi-Core CPUs Using Java 8, 2016, IEEE

[3] Kazuaki Ishizaki, Akihiro Hayashi, Gita Koblents, Vivek Sarkar, Compiling and Optimizing Java 8 Programs for GPU Execution, 2015, IEEE

[4] Bogdan Oancea, Ion Gh. Rosca, Tudorel Andrei, Andreea Iluzia Iacob, Evaluating Java performance for linear algebra numerical computations, 2011, ScienceDirect

[5] Kuo-Yi Chen, J. Morris Chang, Ting-Wei Hou, Multithreading in Java: Performance and Scalability on Multicore Systems, 2011, IEEE

[6] "JMH", 2017, Oracle Corporation and/or its affiliates, http://openjdk.java.net/projects/code-tools/jmh/

[7] "JMH - Java Microbenchmark Harness", 2015, Jakob Jenkov, http://tutorials.jenkov.com/java-performance/jmh.html

[8] Guillermo L. Taboada, Sabela Ramos, Roberto R. Expósito, Juan Touriño, Ramón Doallo, Java in the High Performance Computing arena: Research, practice and experience, 2013, ScienceDirect

[9] Satoshi Matsuoka, Shigeo Itou, Towards performance evaluation of high-performance computing on multiple Java platforms, 2001, ScienceDirect

[10] "Collections", 2014, Oracle, Inc, http://docs.oracle.com/javase/tutorial/collections/intro/index.html

[11] "Paralellism", 2014, Oracle,Inc, http://docs.oracle.com/javase/tutorial/collections/streams/parallelism.html

[12] "Reduction", 2014, Oracle, Inc, http://docs.oracle.com/javase/tutorial/collections/streams/reduction.html

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 148-154
Wysłane: 2017-06-17
Przyjęte: 2017-06-21

Analiza jakościowa sygnałów telemetrycznych w oparciu o dane zgromadzone przez dedykowaną aplikację mobilną

Damian Wasąg*, Małgorzata Plechawska - Wójcik
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Abstrakt. Artykuł prezentuje analizę wartości pulsu człowieka w sytuacji wysiłku fizycznego oraz odpoczynku z podziałem na dwie grupy wiekowe oraz płeć. Pomiary wykonano przy użyciu aplikacji mobilnej oraz ciśnieniomierza nadgarstkowego. Badano dokładność pomiaru telefonem względem ciśnieniomierza oraz istotność statystyczną różnic pomiędzy grupami wiekowymi, płcią oraz wykonywaną czynnością. Niedokładność pomiarowa waha się od 4 do 12 %. Aplikacja mobilna dokładnej odwzorowuje puls podczas odpoczynku.

Słowa kluczowe: puls, tętno, aplikacja mobilna

*Autor do korespondencji.

Adres e-mail: wasag.damian@gmail.com

Qualitative analysis of telemetry signals based on data gathered by dedicated mobile application

Damian Wasąg*, Małgorzata Plechawska - Wójcik
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The paper presents the analysis of the human heartbeat value under the physical exercise and rest condition. The test group was divided into four groups regarding age and sex. Measurements were performed using a mobile application and a blood pressure monitor. The phone's accuracy with respect to the blood pressure gauge was measured and the statistical significance of differences between age groups, sex and activity was also computed. Measurement inaccuracy varies from 4 to 12%. The mobile application accurately reproduces the pulse during rest.

Keywords: pulse; heart rate; mobile application

*Corresponding author.

E-mail address: wasag.damian@gmail.com

1. Wprowadzenie

1.1. Badania literaturowe

Sygnał telemetryczny reprezentuje informację, która została zmierzona oraz jest przesyłana do innego modułu. De la Cruz w artykule [1] opisuje w jaki sposób sygnały telemetryczne są wykorzystywane do pozyskiwania danych biomedycznych pacjentów. Szczególną uwagę poświęca opisowi znaczenia częstotliwości pracy serca w kontekście wcześniejszego diagnozowania komplikacji zdrowotnych.

W następnej pracy [2] udowodniono, sygnału telemetryczny ułatwia badanie pacjenta. Autor zastosował zminiaturyzowany implant. Jego zadaniem jest monitorować czynności życiowe osoby badanej. Urządzenie nie zaburza normalnego funkcjonowania pacjenta oraz umożliwia uzyskanie informacji, np. o aktualnym ciśnieniu krwi.

Töpfer w swojej pracy badawczej [3] wykorzystuje telemetrię do przeprowadzenia terapii resynchronizującej. Opiera się ona na leczeniu wysokiej niewydolności serca. Badacz wprowadza do serca dwie elektrody, które mają za zadanie zapewnić równomierną pracę organu przez pobudzanie komór serca w odpowiednich odstępach czasu. Podkreśla, że szeroko zakrojona analiza otrzymanych wyników pozwala uzyskać szczegółową informację o zaburzeniach pracy konkretnych elementów serca.

Badanie tętna najczęściej opiera się na szacowaniu liczby uderzeń serca na minutę. Ponadto można badać również siłę uderzeń. Wówczas sytuacja pożądana to taka, w której różnica pomiędzy siłą kolejnych uderzeń oscyluje wokół zera. Przedmiotem badań może być również wypełnienie tętnicy oraz napięcie związane z ciśnieniem krwi. W tej pracy naukowej zwrócono uwagę przede wszystkim na częstotliwość uderzeń serca.

Warto zadać sobie pytanie, jak dokonać pomiaru za pomocą telefonu komórkowego? Odpowiedź ułatwia analiza artykułu [4], który obrazuje badanie tętna u biegaczy przy pomocy specjalnie stworzonej do tego celu aplikacji. Dzieło przedstawia jak wyliczyć oraz z jakich właściwości sygnału należy skorzystać, aby otrzymać wynik pulsu. Autor nie uwzględnił niedokładności pomiaru, gdyż nie jest on

porównany z żadnym powszechnie stosowanym urządzeniem badającym puls. Mimo wszystko powyższa praca jest charakterystyczna pod względem prezentacji wyników. Prezentuje pracę serca pod wpływem wysiłku fizycznego przy pomocy wykresów.

Popularność aplikacji o charakterze medycznym jest stosunkowo wysoka. W artykule [5] przedstawiono przegląd dostępnych w 2015 roku aplikacji na smartfony z systemem Android, które ułatwiają monitorowanie oraz rejestrowanie sygnałów biomedycznych pacjentów. Na podstawie jednej z tabel w powyższym artykule można wywnioskować, że zainteresowanie tego typu aplikacjami jest stosunkowo wysokie. Sklep Google odnotował nawet 500000 instalacji jednej z nich. W swoich rozważaniach autor zaznaczył, że rozwój powinien zmierzać nie tylko w kierunku zmniejszenia błędu pomiarowego, ale również w stronę automatyzacji diagnozowania zaburzeń sercowo - naczyniowych.

Eksperyment pomiaru pulsu można przeprowadzić przy użyciu odpowiedniego czujnika [6]. Autor zaznaczył, że proces badania opiera się o technikę fotometryczną. Zmiany pulsu są wykrywane za pomocą detektora, który wykorzystuje światło podczerwone do śledzenia objętości przepływu krwi w tkankach. Twórca zaleca przeprowadzenie eksperymentu na końcówkach palców lub płatach usznych, gdyż w tych miejscach krew ma największe stężenie. W związku z tym błąd pomiarowy będzie niższy.

Technika fotometryczna jest powszechnie wykorzystywana w aplikacjach o charakterze medycznym. Przykładem jest artykuł [7], który prezentuje przetwarzanie obrazu paska testowego do pomiaru glukozy we krwi. Na podstawie przeprowadzonych badań [8], można stwierdzić, że metoda jest bezpieczna. Reprezentuje doskonałą, nieinwazyjną alternatywę dla badań obarczonych ryzykiem wystąpienia komplikacji. Szacowanie tętna w oparciu o nasycenie barw wykorzystał również Sukaphat [9]. Autor wyróżnił się sposobem przeprowadzenia pomiaru. Zrezygnował z dodatkowego czujnika. Jego miejsce zajęła soczewka aparatu wbudowana w telefon komórkowy. Warto wspomnieć, że to rozwiązanie wymaga podświetlenia obszaru ciała za pomocą diody LED wbudowanej w telefon. W swoich rozważaniach badał poziom tętna u 10 osób pomiędzy 20, a 22 rokiem życia w pozycji stojącej. Twórca nie dzielił grupy badawczej pod względem wysiłku fizycznego oraz płci. Ograniczył również czas trwania badania do 10 sekund. Aplikacja mobilna wykorzystana do badania opiera szacowanie poziomu tętna o nasycenie barw RGB. Autor zaprezentował histogramy nasycenia barw, z których wynika, że kanał czerwony najdokładniej odwzorowuje zmiany poziomu tętna. Badania dowodzą, że rozbieżność wyników pomiędzy danymi uzyskanymi z aplikacji mobilnej, a wynikami EKG waha się w okolicach 0,57% .

Na rezultat pomiaru ma wpływ również wybrany do badania smartfon. Dowodzi tego Guede – Fernández [10]. Autor uzyskał niższe wartości błędu dla badań przeprowadzonych przy użyciu telefonu Samsung S5 niż dla Motoroli MX. Telefony różnią się przede wszystkim lampą błyskową i rozdzielczością aparatu. Samsung wyposażony jest w aparat 16 Mpx i standardową lampę błyskową, a Motorola posiada 13 Mpx oraz dwustopniową lampę błyskową.

W powyższych artykułach czas badania był ograniczony. Przykładem urządzenia realizującego pomiar w sposób ciągły są okulary pomiaru pulsu [11]. Zasada ich działania jest podobna do aplikacji mobilnej. Również wykorzystują podświetlenie za pomocą diody LED i detektora, która umożliwia zbieranie informacji. Różnica polega na tym, że proces badawczy opiera się na przepływie krwi przez tętnicę twarzową. Autorzy zaznaczyli, że ruch głową oraz mimika twarzy prowadzi do zakłóceń dokładności badania.

Pomiar pracy serca jest istotnym elementem urządzeń reprezentujących koncepcję Internet of Thigs. Prezentuje to jeden z artykułów [12]. Ukazano tam urządzenie, które wysyła dane do telefonu komórkowego. Aplikacja przetwarza rezultaty pomiaru i prezentuje je w formie wykresu podobnego do EKG. Oprócz tego, dane gromadzone są na zewnętrznym serwerze tak by specjalista zajmujący się leczeniem pacjenta, mógł w dowolnym momencie interpretować wyniki.

Lucena w swoim artykule [13] zwrócił uwagę na fakt, że rozwój aplikacji mobilnych o charakterze medycznym ma szczególne znaczenie dla miejsc, gdzie dostęp do specjalistycznej pomocy medycznej jest ograniczony. Twórca zaznacza, że aplikacje ułatwiają wcześniejsze diagnozowanie chorób. Autor zbadał nie tylko tętno. Zastosowane czujniki oraz mikrokontroler umożliwiły mu badanie temperatury ciała oraz kontroli objętości krwi w tkankach. Mimo wszystko eksperyment przeprowadzono tylko na kilku mężczyznach w wieku od 22 do 52 lat. Grupa badawcza w tym przypadku powinna być znacznie większa i obejmować również kobiety.

W żadnej z prac badawczych nie porównywano pomiarów tętna podczas wysiłku fizycznego oraz w sytuacji minimum godzinnego odpoczynku po obciążeniu organizmu. Wyników pomiarowych pulsu nie zestawiano z rezultatami ciśnieniomierza zakładanego na nadgarstek. Pominięto analizę zapotrzebowania na tego typu rozwiązania techniczne. Żaden z autorów przytoczonych prac nie próbował analizować wyników pod kątem równomiernej pracy serca. Ponadto, nie badano jednocześnie wpływu płci na puls w dwóch grupach wiekowych.

1.2. Hipoteza badawcza

Niniejszy artykuł stawia tezę, że istnieje możliwość realizacji analizy tętna człowieka w oparciu o dane zgromadzone przez dedykowaną aplikację mobilną. Nie ulega wątpliwości, że pomiar obarczony jest błędem. W związku z tym warto zadać kilka pytań. Przede wszystkim, jakim przekłamaniem pomiarowym obarczona jest metoda badawcza, której użyto w tych badaniach. W jakim stopniu otrzymane wyniki będą miarodajnym źródłem informacji? Czy aplikacja mobilna oszacuje puls z jednakowym marginesem błędu dla różnych czynności? Czy rozwój tego typu aplikacji jest potrzebny z punktu widzenia użytkowników? Jak zmienia się liczba skurczów serca dla aplikacji mobilnej, a jak dla ciśnieniomierza wyposażonego

w funkcję pomiaru pulsu? Badania należy przeprowadzić w różnych grupach wiekowych oraz pod wpływem różnych czynności, takich jak odpoczynek oraz wysiłek fizyczny. Wszystkie te pytania zależą od jednej zasadniczej kwestii: czy aplikacja z akceptowalnym poziomem błędu umożliwi oszacowanie pulsu?

2. Materiały i metody

2.1. Sygnały telemetryczne

Pojęcie określane jest mianem mierzalnej wielkości fizycznej, która może się zmieniać w przedziałach czasowych. Oprócz tego sygnał telemetryczny jest silnie związany z telekomunikacją, gdyż opiera się na przesyłaniu różnych pomiarów na różne odległości. W związku z tym reprezentuje informację, która została w pewien jasno określony sposób zmierzona oraz jest przesyłana do innego modułu. Przesyłanie może odbywać się drogą przewodową lub bezprzewodową. Uzyskane wartości mogą być odwzorowane w sposób:

- analityczny,
- liczbowy,
- graficzny.

2.2. Tętno

Szczeklik w jednej ze swoich prac [14] stwierdza, że tętno jest rytmiczna zmianą szerokości tętnic, która odpowiada skurczom serca. Są one uwarunkowane wtłaczaniem porcji krwi do aorty w lewej komorze organu. Analizując jeden z artykułów [15] widać, że zasadnicze znaczenie na wartość pulsu ma wiek. W powyższej pracy przebadano 12 517 osób w wieku od 25 do 87. Jednym z elementów badań było poznanie wartości pulsu u podmiotów badawczych. Okazało się, że osoby starsze miały niższy puls niż uczestnicy badania reprezentujący młodsze pokolenie. W porównania z innym artykułem [16], który miał na celu między innymi detekcję wartości tętna u dzieci w wieku od 2 miesięcy do 18 lat widać już znaczącą różnice, gdyż wyniki oscylowały od 80 do nawet 140 uderzeń serca na minutę. Na podstawie powyższych artykułów sporządzono zestawienie, które przedstawia średnią wartość pulsu dla przedstawicieli różnych grup wiekowych (Tabela 1).

Tabela 1. Wiek, a liczba skurczów serca

Lp	Wiek	Liczba uderzeń na minutę
1	płód	110-115
2	niemowlę	130
3	dziecko	100
4	młodzież	85
5	osoba dorosła	70
6	osoba starsza	60

Liczba uderzeń serca nie zależy wyłącznie od wieku. Wpływ na nią mają często inne niezależne czynniki. Między innymi: płeć, kondycja, przebyte choroby, ogólny stan zdrowia, emocje, pora dnia, w której dokonany zostanie pomiar czy nawet od ułożenia ciała.

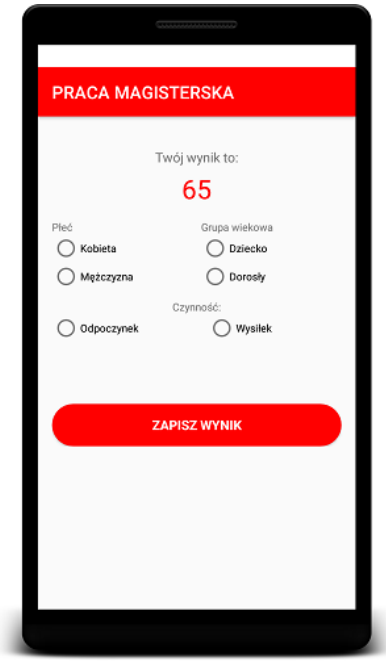
3. Aplikacja mobilna

Dedykowana aplikacja mobilna jest przeznaczona dla urządzeń z systemem Android 6.0 Marshmallow. Jej głównym zadaniem jest oszacowanie wartości tętna oraz zapisanie otrzymanego wyniku. Realizacja tego przedsięwzięcia wymagała deklaracji w pliku Manifest.xml wykorzystania diody LED oraz kamery (Przykład 1).

Przykład 1. Deklaracja uprawnień aplikacji

```
<uses-feature android:name="android.hardware.camera"/>
<uses-feature android:name="android.hardware.camera.flash"/>
<uses-permission android:name="android.permission.CAMERA"/>
```

W odpowiednim miejscu w kodzie aplikacji dokonano inicjalizacji obiektu kamery oraz diody. Następnie w metodzie obsługującej przechwytywanie obrazu z kamery dostarczonej przez klasę adaptera kamery wywoływano funkcje, która na podstawie przekazanej tablicy bajtów obrazu wyliczała nasycenie kanału czerwonego. Otrzymany wynik porównywano z wcześniejszymi rezultatami. Odnotowanie wzrostu powoduje zwiększenie oszacowanej wartości o 1. Pomiar trwa 40 sekund. Wartość liczbowa tętna szacowana jest na podstawie wszystkich klatek zdjęciowych od momentu rozpoczęcia badania do jego zakończenia. Ponadto zapamiętywane są wyniki w odstępach jednej sekundy, które wykorzystywane są tylko w celach prezentacyjnych (Rys. 2). Do analizy wyników brany jest pod uwagę rezultat uzyskany z pomiaru na przestrzeni 40 sekund.



Rys. 1. Formularz zapisu wyniku

Po otrzymaniu wyniku konieczne było jego zapisanie. Na rysunku 1 przedstawiono formularz, w którym osoba badana zobowiązana była wybrać swoją płeć, grupę wiekową oraz czynność w jakiej dokonano pomiaru. Po kliknięciu przycisku zapisz wynik do wbudowanej bazy danych sqlite dodawany jest rekord reprezentujący zrealizowany pomiar oraz wszystkie wyniki jednosekundowych rezultatów. Dzięki temu możliwa jest prezentacja zmian pulsu podczas badania.

Rys. 2. Szczegółowe informacje o pomiarze

4. Procedura badawcza

Badania zostały przeprowadzone przy pomocy dwóch odrębnych urządzeń. Pierwsze to ciśnieniomierz nadgarstkowy salva tec MD 14942 (Rys. 3), który oprócz pomiarów ciśnienia tętniczego, w czasie 35 sekund, wykonuje również banie pulsu. Producent zaznacza, że maksymalny błąd pomiarowy urządzenia oscyluje wokół 5%.

Rys. 3. Ciśnieniomierz z funkcją mierzenia pulsu

Wyniki z tego urządzenia będą porównywane z rezultatami uzyskanymi z drugiego niezależnego pomiaru, który zostanie zrealizowany przy pomocy telefonu HTC Desire 825 (Rys. 4) z zainstalowaną dedykowaną aplikacją mobilną przeznaczoną do tego celu. Smartfon wyposażony jest w system operacyjny Android 6.0 Marshmallow. Do eksperymentu wykorzystano aparat umieszczony na tylnej obudowie, gdyż charakteryzuje się dokładniejszymi parametrami: 13 Mpx matrycą o rozdzielczości 4128 x 3096 px oraz standardową lampą błyskową.

Rys. 4. Komponenty niezbędne do prawidłowego działa aplikacji mobilnej

Grupa badawcza została wybrana w sposób losowy oraz podzielona na cztery podgrupy pod względem: wieku (dorośli i dzieci) oraz płci. Dzieci reprezentowały osoby w wieku 7 lat, a dorośli to osoby 23-letnie. Na każdą podgrupę przypadało po 5 osób. W sumie zostało przebadanych 20 osób. Badanie odbywało się w temperaturze pokojowej w pomieszczeniu. Procedura badawcza została podzielona na dwa niezależne pomiary. Pierwszy wykonany został bezpośrednio po wysiłku fizycznym, drugi natomiast po upływie około godziny w sytuacji odpoczynku. Wysiłek fizyczny obejmował bieg na dystansie 50 metrów w jak najkrótszym czasie. Natomiast odpoczynek polegał na tym, że badany godzinę czasu spędził siedząc na krześle w wygodnej dla niego pozycji. Badanie odbywało się w pozycji stojącej. Osoba badana na nadgarstku prawej ręki miała założony ciśnieniomierz (Rys. 6). Jednocześnie palcem wskazującym tej samej ręki zakrywała cała soczewkę aparatu wraz diodą na czas 40 sekund (Rys 5). Z kolei na ekranie telefonu oraz ciśnieniomierza ukazywał się wynik pomiaru.

Rys. 5. Ułożenie palca wskazującego na diodzie LED oraz soczewce aparatu

Rys. 6. Pomiar przy użyciu ciśnieniomierza oraz telefonu

5. Analiza wyników

Tabela 2. Zestawienie wyników pomiarowych dla grupy wiekowej dzieci w sytuacji wysiłku fizycznego

		Wysiłek fizyczny		
		kobieta	mężczyzna	razem
Aplikacja mobilna	mediana	111	111	111
	dominanta	111	brak	111
	średnia arytmetyczna	110,4	111	110,7
	wariancja	3,04	13,2	16,42
	odchylenie standardowe	1,74	3,63	4,05
Ciśnieniomierz	mediana	125	125	125
	dominanta	124	125	125
	średnia arytmetyczna	125,6	125	125,3
	wariancja	2,64	11,2	7,01
	odchylenie standardowe	1,62	3,35	2,65
średni błąd bezwzględny		15,2	14	14,6
średni błąd względny		12,08%	11,21%	11,65%

Tabela 3. Zestawienie wyników pomiarowych dla grupy wiekowej dzieci w sytuacji odpoczynku

		Odpoczynek		
		kobieta	mężczyzna	razem
Aplikacja mobilna	mediana	94	93	93
	dominanta	94	93	94
	średnia arytmetyczna	91,4	91,8	91,6
	wariancja	13,44	6,16	9,7
	odchylenie standardowe	3,67	2,48	3,11
Ciśnieniomierz	mediana	91	89	90
	dominanta	92	89	89
	średnia arytmetyczna	90,8	89,6	90,2
	wariancja	1,36	0,64	1,36
	odchylenie standardowe	1,17	0,8	1,17
średni błąd bezwzględny		4,2	3,8	4
średni błąd względny		4,63%	4,24%	4,44%

W tabelach 1 i 2 zawarto zbiorcze zestawienie uzyskanych wyników dla grupy wiekowej dzieci w sytuacji wysiłku fizycznego oraz odpoczynku. Analizując wyniki widać przede wszystkim, że średnia arytmetyczna z uzyskanych pomiarów, bez względu na płeć i sposób w jaki dokonano badania, jest wyższa w sytuacji wysiłku fizycznego. Oprócz tego można zauważyć, że zróżnicowanie wyników dla urządzenia pomiarowego na nadgarstek jest niższe, bez względu na to, czy analizujemy rezultaty konkretnej płci, czy nie uwzględniamy tego kryterium. Należy zaznaczyć, że średni błąd aplikacji mobilnej względem ciśnieniomierza jest większy o około 7% w sytuacji wysiłku fizycznego. Uzyskane wyniki z wysiłku fizycznego i odpoczynku w grupie wiekowej dzieci zbadano testem t - Studenta na istotność różnic statystycznych pomiędzy płciami oraz urządzeniami pomiarowymi. Do analizy statystycznej wybrano poziom istotności równy p = 0,01. Na podstawie uzyskanych wyników autor stwierdza, że płeć ma marginalne znaczenie na rezultaty badań. Natomiast różnice pomiędzy wynikami z aplikacji mobilnej i ciśnieniomierza są istotne statystyczne w sytuacji wysiłku fizycznego w tej grupie wiekowej.

Tabela 4. Zestawienie wyników pomiarowych dla grupy wiekowej dorosłych w sytuacji wysiłku fizycznego

		Wysiłek fizyczny		
		kobieta	mężczyzna	razem
Aplikacja mobilna	mediana	89	76	89
	dominanta	89	brak	89
	średnia arytmetyczna	90,6	75,6	83,1
	wariancja	5,84	3,44	50,6
	odchylenie standardowe	2,42	1,85	7,11
Ciśnieniomierz	mediana	98	84	92
	dominanta	98	84	98
	średnia arytmetyczna	99	84,4	91,7
	wariancja	1,6	1,04	54,17
	odchylenie standardowe	1,26	1,02	7,36
średni błąd bezwzględny		8,4	8,8	8,6
średni błąd względny		8,5%	10,43%	9,46%

Tabela 5. Zestawienie wyników pomiarowych dla grupy wiekowej dorosłych w sytuacji odpoczynku

		Odpoczynek		
		kobieta	mężczyzna	razem
Aplikacja mobilna	mediana	69	62	66
	dominanta	68	61	61 i 68
	średnia arytmetyczna	69,2	62,2	65,7
	wariancja	1,36	1,36	13,61
	odchylenie standardowe	1,17	1,17	3,69
Ciśnieniomierz	mediana	72	65	69,5
	dominanta	72	65	72 i 65
	średnia arytmetyczna	72,4	65,6	69
	wariancja	1,04	1,84	13
	odchylenie standardowe	1,02	1,36	3,61
średni błąd bezwzględny		3,2	3,4	3,3
średni błąd względny		4,42%	5,17%	4,8%

Analizując wyniki z tabel 3 i 4, które przedstawiają zbiorcze zestawienie rezultatów pomiarowych dla grupy wiekowej dorosłych, można zauważyć, że płeć wpływa na wyniki. W obu czynnościach średnia wartość pulsu u kobiet była wyższa niż u mężczyzn. Różnica pomiędzy płciami jest najlepiej widoczna w przypadku wysiłku fizycznego. Średnia wartość dla kobiet uplasowała się na poziomie 90,6, zaś u mężczyzn osiągnięto wartość 75,6 uderzeń serca na minutę. Analizując zmiany niedokładności pomiaru w dwóch różnych sytuacjach, widać podobieństwo do wyników dzieci. U dorosłych również odnotowano wyższy błąd dla wysiłku fizycznego. Średnia wartość niedokładności, w tej czynności osiąga 8,6 uderzeń serca, natomiast w sytuacji odpoczynku jest to 3,3. Powyższa różnica została uwidoczniona za pomocą błędu względnego, który podczas wysiłku fizycznego wynosił 9,46% a dla odpoczynku 4,8%. Wyniki dorosłych zbadano

testem t – Studenta na istotność różnic statystycznych pomiędzy płciami oraz urządzeniami pomiarowymi. Otrzymano rezultaty, na podstawie których autor stwierdza że zarówno płeć jak i urządzenie pomiarowe ma istotny wpływ na wartość pulsu zarówno w sytuacji odpoczynku jak i wysiłku fizycznego.

Porównując wyniki dorosłych i dzieci, widać, że wraz z wiekiem puls osiąga niższe wartości. Odnotowano to zarówno za pomocą aplikacji mobilnej, jak i urządzenia pomiarowego. Dla każdego z podmiotów badawczych uzyskano puls wyższy w sytuacji wysiłku fizycznego. Różnice pomiędzy płciami są widoczne tylko dla grupy wiekowej dorosłych. U dzieci nie widać znaczących rozbieżności tego kryterium. Badając dokładność odwzorowania wyników przez aplikacje mobilną, widać, że aplikacja osiąga błąd pomiarowy na niższym poziomie dla sytuacji odpoczynku niż wysiłku fizycznego. Ponadto analizując niedokładność z punktu widzenia grupy wiekowej, rezultaty są dokładniejsze dla dorosłych. U dzieci średni błąd bezwzględny podczas wysiłku fizycznego wynosił aż 14,6 a dla dorosłych w identycznej sytuacji zaledwie 8,6 uderzenia serca na minutę. Analizując wyniki tylko na podstawie liczności przedziałów, widać wyraźnie, że aplikacja mobilna charakteryzuje się większą niedokładnością niż urządzenie pomiarowe.

6. Wnioski

Powyższy artykuł ma na celu analizę zmian pulsu w sytuacji wysiłku fizycznego oraz odpoczynku dla dwóch grup wiekowych: dorosłych i dzieci. W interpretacji przeprowadzonych badań uwzględniono wyniki rozróżniając płeć badanych osób. Oprócz tego oszacowano na ile istotny jest błąd pomiaru dla wyników z aplikacji mobilnej i ciśnieniomierza.

Podczas przeprowadzania badań oraz ich późniejszej analizy nie uwzględniono kilku czynników, które mogą również mieć wpływ na wynik badania. Należą do nich:

- stan emocjonalny,
- różny poziom zmęczenia,
- stopień niewyspania,
- temperatura ciała osób badanych.

Na podstawie uzyskanych rezultatów można stwierdzić, że aplikacja mobilna dokładniej odwzorowuje pomiary u osób dorosłych niż u dzieci. Oprócz tego kluczowe znaczenie ma tutaj fakt, w jakiej czynności dokonano pomiaru. Szacowanie pulsu obarczone jest niższą niedokładnością w sytuacji odpoczynku niż wysiłku fizycznego. Dotyczy to zarówno dorosłych jak i dzieci. W związku z tym metoda pomiaru pulsu nie jest odpowiednia dla sportowców.

Badanie obarczone jest zbyt dużą niedokładnością, która zakłóca pomiar. Przedstawiona metoda badawcza jest wrażliwa na nadmierny nacisk na soczewkę aparatu. Niestosowanie się do procedury badawczej prowadzi albo do przerwania badania poprzez brak wykrycia zmian pulsu lub zawyżonego wyniku.

Prezentowana metoda badawcza mimo wad posiada również kilka istotnych zalet. Przede wszystkim jest tanim sposobem na oszacowanie wartości pulsu, gdyż pomiar może być zrealizowany przy użyciu telefonu. Oprócz tego metoda jest nieinwazyjna. Nie powoduje żadnych niekorzystnych zmian w organizmie badanych osób. Ponadto według autora aplikacja nadaje się do kontroli pulsu do własnych potrzeb dla osób zdrowych.

Literatura

[1] B. De la Cruz, „Transmission of ECG signals with android mobile system via bluetooth," w *Health Care Exchanges (PAHCE)*, Vina de Mar, Chile, 2015.

[2] J. Weidenmüller, „Telemetric multi-sensor system for medical applications - The approach," *Technisches Messen,* styczeń 2017.

[3] A. Töpfer, „Novel telemetric signal averaging ECG approach to determine electrical atrial and ventricular conduction delays in implantable cardioverter defibrillator patients," *Biomedizinsche Technik,* pp. 6 - 9 , 30 wrzesień 2012.

[4] K. J. Hunt i A. J. Hunt, „Feedback control of heart rate durning outdoor running: A smartphone implementation," *Biomedical Signal Processing and Control,* pp. 90-97, kwiecień 2016.

[5] S. Ouhbi, „Mobile personal health records for cardiovascular patients," w *Third World Conference on Complex Systems (WCCS)*, Marrakech, Maroko, 2015.

[6] N. A. Khan, „Real Time Monitoring of Human Body Vital Signs using Bluetooth and WLAN," *International Journal of Advanced Computer Science and Applications,* pp. 210 - 216, październik 2016.

[7] S. Singhal, „Smartphone-based colorimetric detection to measure Blood Glucose Levels," w *Eighth International Conference on Contemporary Computing (IC3)*, Noida, Indie, 2015.

[8] O. I. Bilyy, „Noninvasive Optical Sensor for Tissue Spectroscopic," w *European Conference on Biomedical Optics*, Munch, Niemcy, 2009.

[9] S. Sukaphat, „Heart rate measurement on Android platform," w *13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, Chiang Mai, Tajlandia, 2016.

[10] F. Guede – Fernández, „Real time heart rate variability assessment from Android smartphone camera photoplethysmography: Postural and device influences," w *37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Milan, Włochy, 2015.

[11] N. Constant, „Pulse-Glasses: An unobtrusive, wearable HR monitor with Internet-of-Things functionality," w *12th International Conference on Wearable and Implantable Body Sensor Networks*, Cambridge, USA, 2015.

[12] G. Xiaoqiang, „An ECG Monitoring and Alarming System Based On Android Smart Phone," *Communications and Network ,* styczeń 2013.

[13] S. E. Lucena, „ECG monitoring using Android mobile phone and Bluetooth," w *International Instrumentation and Measurement Technology Conference*, Piza, Włochy, 2015.

[14] A. Szczeklik, Choroby wewnętrzne. Przyczyny, rozpoznanie i leczenie, tom I, Medycyna Praktyczna, 2005.

[15] H. Tomiyama, „Influences of age and gender on results of noninvasive brachial-ankle pulse wave velocity measurement - A survey of 12 517 subjects," *Atherosclerosis,* marzec 2003.

[16] A. Iliff i A. Lee, „Pulse rate, respiratory rate and body temperature of children between two months and eighteen years of age," *Child Development,* pp. 237 - 245, grudzień 2010

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 155-158
Wysłane: 2017-06-19
Przyjęte: 2017-06-23

Metoda docinania faz ruchu podczas wiosłowania na ergometrze na podstawie danych trójwymiarowych

Kacper Kowalczyk*, Maria Skublewska - Paszkowska
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule przedstawione zostały metody modyfikacji danych motion capture w formacie C3D oraz analiza modyfikacji danych przy pomocy autorskiej aplikacji. Aplikacja służy do wczytywania plików w formacie C3D z zarejestrowanym ruchem i automatycznego wycięcia powtarzających się, podobnych sekwencji nagranego ruchu. Analizę przeprowadzono pod kątem porównania ze sobą otrzymanych wyciętych faz ruchu. Pod uwagę brane były długości poszczególnych faz, odległości maksymalne i minimalne pomiędzy badanymi markerami oraz porównywane startowe i końcowe klatki ruchu.

Słowa kluczowe: Motion Capture, Plik C3D, Analiza ruchu 3D

*Autor do korespondencji.

Adres e-mail: kowalczyk.kacper1@gmail.com

Method of cutting phases of motion while rowing on the ergometer based on three-dimensional motion

Kacper Kowalczyk*, Maria Skublewska - Paszkowska
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article presents both the methods of data modification of motion capture data in C3D file format, and the analysis of the modification of motion capture data using implemented application. The application is used to load C3D files with recorded motion and to automatic cut of the repeating, similar sequences of recorded motion. The analysis was conducted in terms of comparing received cut phases of motion. Study include cases such as length of particular phases, maximum and minimum distance between examined markers and comparison of start and final motion frames.

Keywords: Motion Capture, C3D File, 3D Motion Analysis

*Corresponding author.

E-mail address: kowalczyk.kacper1@gmail.com

1. Wstęp

Akwizycja ruchu (ang. motion capture) [1] jest techniką odwzorowywania realistycznych ruchów ciała ludzkiego za pomocą animacji komputerowej. Ruch przechwytywany jest za pomocą kilku kamer rejestrujących położenie markerów, które umieszczone są na ciele aktora. Dane zapisywane są następnie do plików w wybranym formacie. Różne metody modyfikacji danych ruchu mogą być stosowane do obróbki danych po wykonaniu nagrania, jak i do późniejszej analizy konkretnego ruchu.

W artykule przedstawiona została metoda modyfikacji danych trójwymiarowych motion capture zapisanych w formacie plików C3D skupiająca się na aspekcie analizy konkretnych ruchów sportowca pływającego na ergometrze. Metoda polega na wycinaniu powtarzających się faz ruchu pływania na ergometrze zarejestrowanego w pliku C3D. Docięte nagrania zostały porównane ze sobą.

2. Format danych C3D

Format danych C3D zapewnia wygodne i wydajne przechowywanie danych trójwymiarowych razem z analogowymi wraz ze wszystkimi powiązanymi parametrami dla pojedynczych próbek pomiarowych [2]. Format plików C3D jest w powszechnym użyciu od 1987 roku.

Projekt pliku C3D był początkowy inspirowany potrzebą stworzenia wygodnego i wydajnego formatu do przechowywania danych biomechanicznych. Format ten przechowuje współrzędne w jednym pliku w trzech wymiarach i dane numeryczne każdej próbki pomiarowej, wraz ze wszystkimi parametrami opisującymi dane. To w dużym stopniu eliminuje potrzebę przenoszenia próbek pomiarowych z dodatkowymi notkami i informacjami, które narażone mogłyby być na oddzielenie ich od wartości, które opisują przy okazji przenoszenia danych.

Format C3D składa się z trzech podstawowych elementów [3]:

- dane – na tym poziomie plik C3D jest po prostu plikiem binarnym, który przechowuje surowe dane trójwymiarowe i analogowe;
- parametry standardowe – domyślne informacje o danych 3D i analogowych wymagające dostępu do tych danych;
- parametry niestandardowe – informacje specyficzne dla poszczególnych wytwórców aplikacji pomiarowych.

Jednym z celów zaprojektowania formatu C3D było ułatwienie użytkownikowi wylistowania, zbadania i modyfikacji każdego parametru zawartego w pliku. Te parametry dały formatowi możliwość przechowywania wielu informacji o danych. W ten sposób format C3D łączy tradycyjne funkcje przechowywania danych z wieloma charakterystykami dla poszczególnych rekordów danych.

3. Dostępne metody modyfikacji danych motion capture

W pracy naukowej *Data-Driven Completion of Motion Capture Data* autorstwa Jana Baumann, Björn Krüger, Arno Zinke i Andreasa Weber [4] opisane zostało zaimplementowanie metody służącej do uzupełniania (oczyszczania) uszkodzonych danych pochodzących z markerów zarejestrowanego ruchu motion capture z plików C3D. Metoda ta jest oparta na przygotowanych wcześniej danych, więc wykorzystuje ruch z bazy danych motion capture, aby utworzyć bazę startową. Dane do bazy danych pobrane zostały z największych dostępnych baz motion capture – Carnegie Mellon University [5], która zawiera 2605 próbek w 6 kategoriach i 23 podkategoriach oraz bazy HDM05 [6], która zawiera ponad 3 godziny systematycznie i dobrze udokumentowanych nagranych danych. Obie bazy zapewniają dane w formacie C3D.

Dane w bazie startowej muszą być porównywalne z nagraniem, które będzie przetwarzane wyżej wymienioną metodą. Jednym, podstawowym założeniem przedstawionej metody jest to, że wszystkie pozy zawarte w bazie startowej, jak również i te przeznaczone do oczyszczenia, zawierają taki sam zestaw markerów. Poza tym, autorzy założyli, że prawidłowe markery – tj. zestaw markerów, dla których założono, że zawierają wiarygodne informacje odnośnie swoich pozycji – są dostępne dla każdej klatki danych przeznaczonych do uzupełnienia.

L. Herda w pracy *Skeleton-based motion capture for robust reconstruction of human motion* [7] opracował technikę śledzenia i rekonstrukcji markerów opartą na szkielecie służącą do wydedukowania pozycji brakujących markerów poprzez użycie informacji kinematycznych dostarczanych przez bazowy szkielet i dane współrzędnych markerów z poprzedzających klatek, które są umieszczone na tym samym segmencie ciała aktora. Ta metoda jest odpowiednia do krótkich przysłonięć pojedynczych markerów, ale zawodzi w przypadku, gdy cały segment markerów jest przysłonięty przez wydłużony okres czasu.

Filtry Kalmana były wykorzystywane w *Robust Optical User Motion Tracking Using a Kalman Filter* autorstwa K. Dorfmuller-Ulhaas [8] do przewidywania trajektorii brakujących markerów. Jednak filtry Kalmana w podstawowych podejściach zawodzą, gdy markerów brakuje przez dłuższy czas lub brak ich przez całe zarejestrowane nagranie ruchu.

L. Li w pracy *Bolero: A principled technique for including bone length constraints in motion capture occlusion filling* [9] proponuje metodę uzupełniania luk w danych markera poprzez zastosowanie systemu liniowo-dynamicznego które uwzględnia odległości pomiędzy markerami. Technika ta wymaga, żeby istniały inne markery w tym samym segmencie, aby umożliwić utworzenie pomiarów odległości pomiędzy markerami.

W ostatnich latach zaproponowane zostały dodatkowe metody służące do oczyszczania danych motion capture oparte na istniejących danych. H. Lou i J. Chai w artykule *Example-based human motion denoising* opisali, w jaki sposób można przefiltrować uszkodzone dane ruchu poprzez wykorzystanie serii przestrzenno-czasowych baz filtrów z wcześniej nagranego ruchu. Wykorzystując ich podejście do problemu oparte na filtrach, mogli zredukować szum, usunąć odchyły (dane odstające) i uzupełnić luki podczas zachowywania przestrzenno-czasowych wzorców filtrowanego ruchu ludzkiego w postaci nienaruszonej. Ich metoda wymaga utworzenia treningowej bazy danych w czasochłonnym etapie, który wyłącznie zawiera ruch podobny do ruchu, który ma zostać oczyszczony. Tak więc, w kontraście do metody przedstawionej w *Data-Driven Completion of Motion Capture Data*, nie jest ta metoda w stanie poradzić sobie z różniącymi się znacznie stylami ruchu równocześnie bez kosztownego preprocessingu.

J. Xiao w *Predicting missing markers in human motion capture using l1-sparse representation* [10] opracował metodę uzupełniania luk poprzez reprezentację niekompletnych poz poprzez liniową kombinację kilku poz ze zbioru testowego. To podejście do tematu, zarówno jak i w *Example-based human motion denoising* wymaga testowego zbioru danych motion capture zawierającego podobne wzorce ruchu (w dodatku od tego samego aktora) jak ruch, który ma zostać oczyszczony. Ponadto, rzetelność tego podejścia do dodatkowych danych niepowiązanych w zbiorze testowym nie została omówiona.

4. Badania

Przeprowadzone zostały badania dotyczące możliwości edycji plików C3D zaimplementowanej aplikacji. W skład badań wchodziło wycinanie powtarzających się ruchów z trajektorii, nazywanych fazami, w celu ich porównania. Do celów badań została zaimplementowana aplikacja.

4.1. Aplikacja

Aplikacja została utworzona w języku C#. Środowiskiem programistycznym wykorzystanym do implementacji było Microsoft Visual Studio Community 2013. Zaimplementowane funkcje służące do przeprowadzonych badań zrealizowane zostały przy pomocy darmowej biblioteki C3D.NET [11], umożliwiającej wczytywanie, modyfikację i przeglądanie zawartości plików C3D.

W skład funkcjonalności aplikacji wchodzi wczytywanie pliku, automatyczne wykrycie i wycięcie faz, wyświetlenie informacji o wyciętych fazach oraz zapisanie każdej fazy do oddzielnego pliku C3D.

4.2. Wycinanie powtarzającej się fazy z pliku

Zaimplementowana została metoda modyfikacji plików C3D polegająca na wycinaniu z pliku powtarzających się faz ruchu. Powodem, dla którego zostały wykonane te badania, jest chęć analizy powtarzających się sekwencji ruchów w zarejestrowanym nagraniu. Przydatne jest to przy analizie nagrań wykonywania czynności sportowych.

W rozpatrywanym przypadku analizie poddany został plik z zarejestrowanym ćwiczeniem na ergometrze. Ćwiczenia na ergometrze składają się z powtarzalnych ruchów więc są bardzo dobrym przykładem, aby wyciąć te powtarzające się ruchy i umożliwić ich jednostkową analizę w celu na przykład zwiększenia osiągów sportowca.

Celem wykonanych badań jest porównanie pięciu wyciętych faz ruchu pływania na ergometrze i wskazania różnic pomiędzy nimi.

Przeprowadzenie badań polegało na wczytaniu pliku C3D i wycięciu pierwszych pięciu faz ruchu względem wskazanego markera. W badanym przypadku był to marker umieszczony na lewej dłoni aktora. Faza wyznaczana jest od klatki, dla której jest najmniejsza odległość od markera statycznego względem wskazanego wcześniej markera, do kolejnej klatki z najmniejszą odległością od markera statycznego. Marker statyczny, czyli taki, który znajduje się przez cały czas w tym samym położeniu. W analizowanym przypadku marker statyczny znajduje się na słupku ergometru.

Wycięte fazy są następnie zapisywane do nowych, oddzielnych plików C3D, aby łatwiej można było analizować konkretną fazę. Wraz z wycięciem faz obliczane są ich parametry, które zostały następnie do tabeli z wynikami. Dane te zostały zebrane w Tabeli 1, w której zapisano:

- numer fazy – numer kolejno wyciętych faz;
- długość – długość wyciętej fazy liczona w klatkach;
- odległość maksymalna – największa odległość pomiędzy badanym markerem, a markerem statycznym umieszczonym na słupku ergometru;
- odległość minimalna – najmniejsza odległość pomiędzy badanym markerem, a markerem statycznym;
- odległość średnia – średnia odległość pomiędzy badanym markerem, a markerem statycznym;
- numer klatki początkowej – numer klatki z oryginalnego pliku, od której zaczyna się konkretna faza;
- numer klatki końcowej – numer klatki z oryginalnego pliku, na której kończy się konkretna faza.

Wycinanie fazy zostało wykonane w oparciu o markera umieszczonego na lewej dłoni, przy palcu wskazującym (*LFIN* według modelu Plug-in Gait [13]*)* człowieka ćwiczącego na ergometrze. Odległości w trójwymiarowym układzie współrzędnych liczone były od markera znajdującego się na słupku ergometru (markera statycznego). Odległość między markerami obliczono według wzoru (1).

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (1)$$

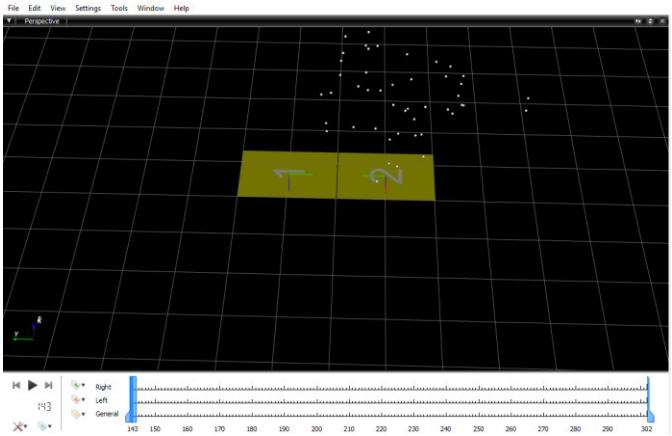
gdzie:

- x_2 – wartość położenia markera LFIN na płaszczyźnie X;
- x_1 – wartość położenia markera statycznego na płaszczyźnie X;
- y_2 – wartość położenia markera LFIN na płaszczyźnie Y;
- y_1 – wartość położenia markera statycznego na płaszczyźnie Y;
- z_2 – wartość położenia markera LFIN na płaszczyźnie Z;
- z_1 – wartość położenia markera statycznego na płaszczyźnie Z.

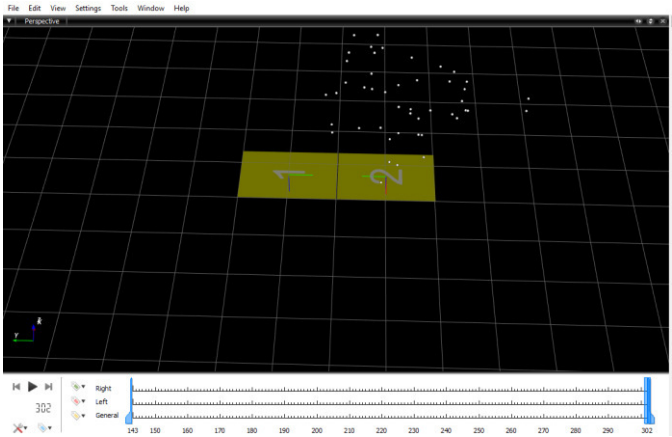
Algorytm działania wycinania faz został przedstawiony w następujących punktach:

1) Podawany jest przez użytkownika marker, dla którego analizowany jest ruch.
2) Ustalana jest wartość środkowa dla trajektorii podanego markera. Wartość ta jest wyliczana jako środek zakresu pomiędzy wartością max a wartością min położenia podanego markera.
3) Liczona jest liczba przecięć wykresu z wartością środkową oraz pobierane są indeksy klatek znajdujących się na przecięciu.
4) Wartości klatek, które są mniejsze od wartości środkowej znajdujące się w zakresie od indeksu przecięcia do kolejnego indeksu przecięcia wstawiane są do tablicy.
5) W każdej tablicy znajdowana jest klatka z najmniejszą wartością i zwracany jest jej indeks.
6) Fazy wybierane są z klatek z zakresów pomiędzy indeksami najmniejszych wartości. Indeks klatki początkowej wszystkich faz oprócz pierwszej jest równy indeksowi klatki końcowej fazy jej poprzedzającej.
7) Następuje wycięcie faz i zapisanie do oddzielnych plików.
8) Zwracane są informacje o wyciętych fazach.

Wizualizacja wyciętych faz przeprowadzona została za pomocą aplikacji Mokka [12]. Na rysunkach 1. i 2. zaprezentowano zrzuty ekranu z Mokki porównujące pierwszą i ostatnią klatkę fazy nr 1.



Rys.1. Zrzut ekranu z programu Mokka przedstawiający pierwszą klatkę Fazy 1



Rys.2. Zrzut ekranu z programu Mokka przedstawiający ostatnią klatkę Fazy 1

Tabela 1. Wyniki badań

Nr fazy	Faza 1	Faza 2	Faza 3	Faza 4	Faza 5
Długość (liczba klatek)	160	154	156	148	156
Odległość maksymalna [mm]	1500,41	1515,86	1518,63	1499,40	1460,67
Odległość minimalna [mm]	363,42	335,43	335,43	330,73	330,73
Odległość średnia [mm]	856,43	836,46	825,28	823,32	798,12
Nr klatki początkowej	143	302	455	610	757
Nr klatki końcowej	302	455	610	757	912

Najdłuższą fazą z poddanych badaniu jest faza nr 1 trwająca 160 klatek, natomiast najkrótszą jest faza nr 4, która trwa 148 klatek. Widać tendencję skracania faz w kolejnych pociągnięciach. Marker LFIN jest najbardziej oddalony od markera statycznego w fazie 3, z kolei najmniejsze oddalenie maksymalne jest dla fazy numer 5. Najmniejsza odległość minimalna jest dla faz nr 4 i 5 – wartości te są sobie równe ponieważ faza 4 kończy się na klatce nr 757, a faza 5 zaczyna się na klatce 757. Największa wartość odległości minimalnej jest dla fazy 1.

5. Podsumowanie

Dotychczasowe badania skupiające się na zagadnieniu modyfikacji danych motion capture, które zostały omówione w niniejszej pracy dotyczyły głównie aspektu korygowania oraz uzupełniania danych zarejestrowanego ruchu. W znacznym stopniu przyczyniły się zatem do pogłębienia wiedzy praktycznej w kwestii modyfikacji danych pochodzących z plików C3D oraz przyspieszyły rozwój prezentowanej problematyki.

Opracowane w niniejszym artykule badania miały na celu rozszerzenie zagadnienia modyfikacji danych motion capture z plików C3D o dodatkowe aspekty, które pozwalają na głębszą analizę samego ruchu, a nie tylko jego jakości technicznej.

Otrzymane rezultaty mogą przyczynić się do analizowania, a w konsekwencji do poprawy jakości ruchowej zawodnika. Dzięki opracowanej metodzie możliwe będzie porównywanie powtarzających się faz ruchu po to, aby pracować nad ich ulepszeniem. W przyszłości może to okazać się niezwykle pomocne dla sportowców, którym szczególnie zależy na poprawie osiąganych wyników w zakresie sprawności fizycznej.

Literatura

[1] Tobon R.: The Mocap Book: A Practical Guide to the Art of Motion Capture, 2010

[2] https://www.c3d.org/, 2016

[3] The C3D File Format User Guide, https://www.c3d.org/pdf/c3dformat_ug.pdf, 2008

[4] Baumann J., Krüger B., Zinke A., Weber A.: Data-Driven Completion of Motion Capture Data, 2011

[5] Carnegie Mellon University Motion Capture Database, http://mocap.cs.cmu.edu/, 2004

[6] Documentation: Mocap Database HDM05, http://resources.mpi-inf.mpg.de/HDM05, 2007

[7] Herda L.: Skeleton-based motion capture for robust reconstruction of human motion, 2000

[8] Dorfmüller-Ulhaas K.: Robust Optical User Motion Tracking Using a Kalman Filter, 2007

[9] Li L., McCann J., Pollard N., Faloutsos C.: BoLeRO: A Principled Technique for Including Bone Length Constraints in Motion Capture Occlusion Filling, 2010

[10] Xiao J., Feng Y., Hu W.: Predicting missing markers in human motion capture using l1-sparse representation, 2011

[11] C3D.NET, https://c3d.codeplex.com/, 2014

[12] Barrea A., Armand S: Biomechanical ToolKit: Open-source framework to visualize and process biomechanical data, 2014

[13] Plug-In Gait Model Details, https://www.vicon.com/downloads/documentation/vicon-documentation/plug-in-gait-model-details, 2017

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 159-166
Wysłane: 2017-06-22
Przyjęte: 2017-06-29

Analiza porównawcza jakości interfejsów mobilnego dostępu do usług wybranych banków

Marcin Kurzyna*, Damian Matysiak, Marek Miłosz
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Niniejszy artykuł jest poświęcony analizie jakości interfejsów wybranych bankowych aplikacji mobilnych. W trakcie badań brano pod uwagę czynniki takie jak bezpieczeństwo aplikacji, dostępność na platformy mobilne, szybkość wykonywania zadań, prostota obsługi, zrozumiałość, oprawa graficzna, komfort pracy, liczba funkcjonalności. W badaniach wykorzystane zostały badania ankietowe, testy korytarzowe, badania eksperckie oraz badania eyetrackingowe. Przeprowadzone eksperymenty ujawniły, że aplikacja T-Mobile Usług Bankowych przewyższa aplikację mBanku w kategoriach zrozumiałości, oraz prostoty obsługi. Aplikacja mBanku została lepiej oceniona pod względem responsywności, liczby funkcjonalności oraz dostępności.

Słowa kluczowe: jakość interfejsu; analiza porównawcza; bankowość mobilna

*Autor do korespondencji.

Adres e-mail: marcin.kurzyna@pollub.edu.pl

A comparative analysis of interface quality of mobile access to the services of selected banks

Marcin Kurzyna*, Damian Matysiak, Marek Miłosz
[a] Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Abstract. This paper is focused on comparison of interface quality of mobile access to the services of selected banks. Main factors, which were used in the comprising process are: safety, availability on the mobile platforms, time and simplicity of task performing, intelligibility, design, comfort, number of functions. Research was made using 4 methods: surveys, corridor tests, expert tests and eyetracking. As a result T-Mobile Banking Services won in three categories: intelligibility, design and simplicity while mBank is better in responsiveness, availability and has more functions to offer.

Keywords: quality of interface; comparative analysis; mobile banking

*Corresponding author.

E-mail address: marcin.kurzyna@pollub.edu.pl

1. Wstęp

W dzisiejszych czasach zauważalny jest duży wzrost powszechności urządzeń mobilnych, takich jak smartphony, tablety, phablety, czy smartwatche. W wykładniczym tempie wzrasta też ich moc obliczeniowa oraz wachlarz oferowanych przez nie funkcjonalności. Coraz częściej urządzenie mobilne zastępuje ludziom przedmioty codziennego użytku, jak np. budzik, notes, kalendarz, itd. Nie dziwi więc fakt, że smartphone może zastąpić również portfel. Szereg banków w celu sprostania oczekiwaniom „społeczeństwa mobilnego" wprowadza do swojej oferty bankowe aplikacje mobilne, które pozwalają w łatwy sposób, przy użyciu urządzenia mobilnego kontrolować stan swojego konta oraz swobodnie obracać środkami pieniężnymi, które są na nim zgromadzone.

Najważniejszym zadaniem aplikacji mobilnej jest spełnianie oczekiwań jej odbiorców. Badając grupy użytkowników mobilnych aplikacji bankowych można w prosty sposób przeanalizować jakość interfejsów tychże aplikacji. Dodatkowe wsparcie specjalistycznej aparatury pomaga wychwycić chybienia popełnione w procesie projektowania aplikacji i przyczynić się do ich sprostowania.

Istnieje szereg metod badania jakości interfejsów aplikacji mobilnych. Do najbardziej popularnej formy badań można zaliczyć badania ankietowe, które pozwalają na wysuwanie hipotez odnośnie całej populacji na podstawie grupy reprezentatywnej będącej ich częścią. Kolejnym typem badań są tak zwane testy korytarzowe. Idea tego typu badań polega na testowaniu interfejsu na przypadkowych osobach, zlecając tymże osobom wykonanie kilku prostych zadań. Pozwala to na bezpośrednie śledzenie zachowań użytkowników, którzy często korzystają z aplikacji po raz pierwszy. Tego typu testy umożliwiają szybkie wychwycenie słabych stron aplikacji bez znaczącego nakładu czasu i pieniędzy. Trzecim rodzajem badań są tzw. badania eksperckie, które polegają na przeprowadzeniu wywiadu z osobami, które mają duże doświadczenie w korzystaniu z aplikacji. Czwartym typem możliwego do przeprowadzeniu eksperymentu jest eyetracking, nazywany również badaniem okulograficznym. Eyetracker jest to specjalistyczny sprzęt pozwalający na śledzenie ruchu gałek ocznych użytkowników. Za pomocą okulografu można dowiedzieć się, które elementy interfejsu absorbują największą uwagę użytkownika. Urządzenia daje możliwość sporządzenia map cieplnych oraz map fiksacji wzroku, które są wygodnym do interpretacji materiałem badawczym.

Dodatkowo eyetracker umożliwia rejestrowanie przebiegu eksperymentu, co umożliwia bardziej wnikliwą analizę.

1.1. Badania literaturowe

Wielu autorów łączy podobne stanowisko w odniesieniu do kluczowych cech interfejsu użytkownika. Ich zdanie nie jest jednak w tym temacie jednogłośne. Badacze uważają różne cechy za mniej, bądź bardziej istotne.

Memon Atif w swojej pracy „*User Interface Design Designing effective interfaces for software systems*" [1] sugeruje, że najważniejszą cechą interfejsu jest zrozumiałość. Jego zdaniem składowe interfejsu powinny być reprezentowane przez elementy znane użytkownikowi z życia codziennego (np. ikona telefonu reprezentująca ekran z informacjami kontaktowymi).

Theo Mendel w pozycji „*The Elements of User Interface Design*" [2] zwraca uwagę, że interfejs powinien przede wszystkim zapewniać użytkownikowi swobodę działania. Mendel twierdzi, że niewłaściwe jest narzucanie jednego scenariusza przypadku użycia. Według autora możliwość mnogość alternatywnych sposobów nawigacji przekłada się na zwiększenie komfortu oraz efektywności pracy użytkownika interfejsu.

Oprócz ogólnych badań jakości interfejsów, znaleźć można również wiele pozycji dostępnych w formie elektronicznej skupionych na analizach interfejsów mobilnych aplikacji bankowych.

Kamil Ormaniec w swoim artykule „*Najlepsze aplikacje bankowe w Polsce – ranking Komórkomat.pl*" [3] zestawia najczęściej wykorzystywane bankowe aplikacje mobilne i tworzy ranking dziesięciu najlepszych. Swój ranking tworzy on w oparciu o różnorodne kryteria. Ormaniec uwzględnia takie czynniki jak wygląd interfejsu, dostępność programów lojalnościowych, czy też liczba różnych metod płatności.

Aaron Marcus w swoim artykule „*Cross-cultural user-interface design for work, home, play, and on the way*" [4] zwraca uwagę, że przy procesie projektowania interfejsów aplikacji aspirujących do osiągnięcia międzynarodowego zasięgu należy zwracać uwagę na różnice kulturowe użytkowników. Podobne zdanie wyraża Marek Miłosz w swoim podręczniku „*Ergonomia Systemów Informatycznych*" [5]. Według autora różnice te mogą przejawiać się chociażby w kierunku odczytywania treści. Dla użytkowników, których językiem ojczystym jest język arabski, odczytywanie komunikatów następuje od strony prawej do lewej, z kolei u ludzi ze wschodniego kręgu kulturowego od góry do dołu. Tego typu niejednoznaczności mogą utrudniać zrozumienie sekwencji czasowych, które mogą pojawić się np. w samouczkach (użytkownik mylnie odczytuje instrukcje od końca do początku).

Artykuł „*Bezpieczeństwo bankowości mobilnej - jak bezpiecznie wykonywać transakcje mobilne*" autorstwa Beaty Szymańskiej [6] szczegółowo opisuje kwestię bezpieczeństwa bankowych aplikacji mobilnych. W tekście tym można znaleźć informacje na temat najczęstszych metod uzyskiwania nieuprawnionego dostępu do konta takich jak: Bug Stagefright, wykorzystującą lukę w oprogramowaniu systemu Android; atak za pośrednictwem protokołu SSDP w systemie Windows 8.1. W artykule znaleźć można również dane na temat zabezpieczeń stosowanych przez banki, takich jak np. obrazek bezpieczeństwa, czy zabezpieczenia biometryczne.

Na podstawie raportu „*BANK.JEST.MOBI 2016"* Moniki Mikowskiej [7] można uzyskać informację na temat złych praktyk, które towarzyszą twórcom mobilnych aplikacji bankowych. Autorka zwraca uwagę na fakt, że niektóre banki wymagają tych samych danych do logowania w aplikacji oraz na stronie internetowej. Mikowska uważa, że brak jest optymalnej metody logowania do aplikacji. Swoje stanowisko autorka opiera o negatywne opinie użytkowników, dla których wprowadzanie haseł składających się często z cyfr oraz liter różnej wielkości jest uciążliwe. Z drugiej strony alternatywny sposób logowania poprzez kod PIN jest przez autorkę uznawany za mało bezpieczny.

Poza bezpieczeństwem, często poruszaną kwestią jest możliwość korzystania z aplikacji przez osoby niepełnosprawne. Karolina Śladkowska w swoim zestawieniu „*5 problemów polskiej bankowości mobilnej*" [8] pisze, że twórcy aplikacji bankowych rzadko wdrażają do swoich produktów narzędzia ułatwiające dostęp osobom niewidomym i niedowidzących takie jak czytniki ekranu. Autorka zwraca również uwagę na trudność kontaktu osób z zaburzeniami słuchu z konsultantem. Wiele aplikacji oferuje tego typu kontakt jedynie na drodze połączenia telefonicznego.

Badając pozycje literaturowe natrafić można na prace stricte dotykające zagadnienia oceny jakości bankowości mobilnej. Przykładem fachowej literatury zajmującej się tym problemem jest artykuł *„Examining the key dimensions of mobile banking service quality: an exploratory study"* [9] autorstwa Minjoon'a Jun'a oraz Sergio Palacios'a. Naukowcy w swoich badaniach analizują mobilne serwisy bankowe na tle siedemnastu kryteriów, wśród których są między innymi: prostota użycia, bezpieczeństwo, świadomość produktu, responsywność oraz ciągły rozwój aplikacji.

Ważnym krokiem w planowaniu badań jest zapoznanie się z metodami analizy wielokryterialnej. W literaturze można znaleźć wiele metod pomocnych w dokonaniu prawidłowej oceny jakości interfejsu. W artykule Luizy Fabisiak oraz Pawła Ziemby pt. „W*ybrane metody analizy wielokryterialnej w ocenie użyteczności serwisów internetowych"* [10] można odnaleźć informacje na temat metody analitycznej hierarchizacji problemu- AHP (ang. *Analytic Hierarchy Process*). Technika ta polega na graficznym budowaniu modelu hierarchii celów, w której ustalany jest oraz dekomponowany problem decyzyjny, co pozwala na opisanie struktury problemu decyzyjnego, co w rezultacie prowadzi do jego dekompozycji. W efekcie tego procesu osiągane zostają cele, które są podzielone w procesie porównawczym pod względem ważności.

Anton Smoliński w pozycji „*Klasyfikacja i porównanie metod klasycznej optymalizacji wielokryterialnej”* [11] wspomina o metodzie sum ważonych. Metoda ta pozwala przekształcić problem wielokryterialny, na jednokryterialny, przez ustalenie wag danych kryteriów porównawczych. Autor wymienia szereg zalet tej metody. Zwraca on uwagę na jej prostotę oraz intuicyjność. Z drugiej Smoliński uczula też czytelników na problemy związane z wykorzystywaniem tej metody. Naukowiec sugeruje, że dobór odpowiednich wag kryteriów musi być poprzedzony solidnym rozeznaniem, jakie choćby można dokonać studiując literaturę przedmiotu.

Bogactwo materiałów bibliograficznych daje solidne podstawy teoretyczne do przeprowadzenia analizy porównawczej jakości interfejsów mobilnego dostępu do usług wybranych banków. Wzorowanie się na pozycjach literaturowych daje możliwość przeprowadzenia rzetelnych badań, dając jednocześnie możliwość zaprezentowania znalezienia nieodkrytych do tej pory ścieżek i zaprezentowania nowatorskich wyników wykonanych eksperymentów.

1.2. Cel badań

Celem pracy było przeprowadzenie analizy porównawczej jakości interfejsów mobilnych aplikacji bankowych na przykładzie aplikacji T-Mobile Usług Bankowych i mBanku. Badania miały ujawnić, które cechy interfejsu użytkownika wpływają w największym stopniu na efektywność wykonywanych przez użytkownika czynności. Przeprowadzone eksperymenty pomogły wykryć zaprojektowane w niewłaściwy sposób elementy interfejsu, pozwoliły określić poziom bezpieczeństwa aplikacji bankowych, co może przełożyć się na poprawę jakości kolejnych wersji aplikacji.

2. Materiały i metody badawcze

2.1. Kryteria porównawcze i ich wagi

Posiłkując się badaniami literaturowymi, szczególnie pozycjami odnoszącymi się do użyteczności, sporządzona została następująca lista kryteriów porównawczych (tab. 1). Kryteriom porównawczym zostały przypisane odpowiednie wagi.

Tabela 1. Wagi kryteriów porównawczych

Kryterium	Waga
K1 Bezpieczeństwo	24
K2 Średni czas realizacji zadania	19
K3 Dostępność	15
K4 Zrozumiałość	13
K5 Komfort pracy	11
K6 Prostota	9
K7 Liczba sposobów dok. płat.	6
K8 Ocena wizualna	3
SUMA	100

Interfejs danej aplikacji zostanie uznany za lepszy jeśli uzyska największą ocenę sumaryczną, która wyznaczana jest za pomocą wzoru (1).

$$k_F = \left[\sum_{i=1}^{8} w_i k_i \right] \tag{1}$$

gdzie:

$$k_i = \frac{1}{10 n_i} \sum_{j=1}^{n_i} m_{ij}, m_{ij} = 1,2,\ldots,10 \tag{2}$$

k_F – ocena sumaryczna (punkty końcowe)

w_i – waga i-tego kryterium

k_i – ocena aplikacji wg i-tego kryterium

n_i – liczba metod badających i-te kryterium

m_{ij} – ocena aplikacji w i-tym kryterium j-tej kategorii (subkryterium)

$[\alpha]$ – zaokrąglenie liczby α do liczby całkowitej

2.2. Opis obiektów badawczych

Obiektami badawczymi były dwie mobilne aplikacje bankowe: mBank oraz T-Mobile Usługi Bankowe. Aplikacja mBank dostępna jest dla użytkowników urządzeń z systemami operacyjnymi Android, iOS, Windows Phone oraz Windows 8, natomiast aplikacja T-Mobile Usługi Bankowe dostępna jest na systemy Android i iOS. Badania zostały przeprowadzone na wersji 3.1.0 w przypadku mBanku oraz 4.1.4 w przypadku T-Mobile Usług Bankowych.

2.3. Opis metod badawczych

Do przeprowadzenia badań jakości interfejsów mobilnego dostępu do usług wybranych banków użyto następujących metod:

1. Badania ankietowe: Badania ankietowe służą do pozyskania opinii od większej liczby osób na dany temat. Badania ankietowe dla mobilnych aplikacji bankowych zostały zrealizowane elektronicznie. Do przygotowania ankiet posłużyło narzędzie „Formularze Google”.
2. Testy korytarzowe: Testy korytarzowe są często wykorzystywane w przypadku niskiego budżetu projektu. Metoda jest jednak używana przez większość projektantów, często w sposób nieświadomy. Polega ona na badaniu odczuć użytkowników w czasie pracy z daną aplikacją. Badanymi są często osoby przypadkowe lub koleżanki czy koledzy z pracy. Testy korytarzowe mogą być przeprowadzane już na wstępnych prototypach aplikacji.
3. Badania eksperckie: Jak sama nazwa wskazuje badania eksperckie przeprowadzane są na grupie ekspertów w danej dziedzinie. Doświadczenie i wiedza osób biegłych w konkretnym temacie przynosi nierzadko wiele cennych

uwag. Eksperci dzięki swojej wprawie mogą dostrzec o wiele więcej szczegółów niż osoby przypadkowe biorące udział w testach korytarzowych.

4. Badania eyetrackingowe, okulografia (ang. *eye tracking)* jest to technika polegająca na śledzeniu ruchu gałek ocznych osoby badanej. Eyetracking stosowany jest w wielu dziedzinach między innymi w psychologii, medycynie czy informatyce. Istnieją dwa rodzaje eyetrackingu: stacjonarny oraz mobilny. W przypadku eyetrackingu mobilnego wykorzystuje się specjalne okulary, zawierające kamery skierowane na oczy osoby badanej. Okulary podłącza się do smartfona, z zainstalowanym, specjalnym oprogramowaniem. Osoba badana (testujący) musi zostać wprowadzona do urządzenia np. poprzez podanie imienia, a następnie poddana procesowi kalibracji przy pomocy specjalnej tablicy z wyznaczonymi punktami.

Każda z metod badawczych pozwala na ocenę różnych zestawów kryteriów zawartych w tab. 1. Przyporządkowanie kryteriów poszczególnym metodom prezentuje tab. 2.

Tabela 2. Kryteria badane danymi metodami

Metoda	Kryterium
Badania ankietowe	K1, K4, K5, K6, K7, K8
Testy korytarzowe	K4, K6, K8
Badania eksperckie	K1, K3, K5, K7, K8
Badania eyetrackingowe	K4, K6

2.4. Metody statystyczne

Często wykorzystywanym narzędziem statystycznym służącym do porównywania dwóch prób jest test t-Studenta. Pozwala on na oszacowanie z góry założonym istotności przedziału, w którym znajduje się dana wielkość. Dane analizowane tą metodą muszą jednak pochodzić z populacji o rozkładzie Gaussa oraz mieć podobną liczebność (zwykle w badaniach społecznych zakłada się, że liczebność jednej próby nie może przekroczyć dwukrotnie liczebności drugiej próby).

Statystyka t-Studenta podana jest wzorem:

$$T = \frac{\bar{x}_1 - \bar{x}_2}{S} \tag{3}$$

gdzie:

$$S = \sqrt{\frac{(n_1 - 1)\hat{s}_1^{\,2} + (n_2 - 1)\hat{s}_2^{\,2}}{n_1 + n_2 - 2}} * \frac{n_1 + n_2}{n_1 n_2} \tag{4}$$

n_m – liczebność prób dla poszczególnych aplikacji, *m=1* dla mBanku, *m=2*- T-Mobile UB

$\bar{x}_m$ – średnia arytmetyczna dla poszczególnych prób

$\hat{s}_m^{\,2}$ – nieobciążony estymator wariancji

Obliczoną statystykę t-Studenta porównuje się ze stablicowanymi wartościami, przy zadanych liczebnościach prób i poziomie istotności.

Wynik testu t-Studenta odczytywany jest w następujący sposób:

- Jeżeli $T \geq t_\alpha^{\,n_1+n_2-2}$, wówczas na poziomie istotności α można stwierdzić, że aplikacja mBanku jest lepsza od aplikacji T-Mobile Usług Bankowych w danej kategorii.
- Jeżeli jest spełniona nierówność $T \leq -t_\alpha^{\,n_1+n_2-2}$, wówczas na poziomie istotności α można stwierdzić, że aplikacja T-Mobile Usług Bankowych jest lepsza od aplikacji mBanku w danej kategorii.
- Jeżeli zachodzi $-t_\alpha^{\,n_1+n_2-2} < T < t_\alpha^{\,n_1+n_2-2}$, wówczas na poziomie istotności α można stwierdzić, że pomiędzy interfejsami nie ma znaczących różnic.

W celu zbadania zbieżności danych z rozkładem normalnym stosuje się metodę Shapiro-Wilka. W przypadku wyników nieposiadających rozkładu zbliżonego do normalnego zostaną zastosowane metody statystyki opisowej.

3. Wyniki

3.1. Badania ankietowe

W przypadku aplikacji mobilnej T-Mobile UB 58% respondentów stanowili mężczyźni, kobiety stanowiły odpowiednio 42% badanych osób. Większość ankietowanych to osoby między 22 a 24 rokiem życia. Wyniki badań, które dotyczą aplikacji mobilnej T-Mobile Usług Bankowych przedstawia tab. 3.

Tabela 3. Wyniki badań ankietowych aplikacji T-Mobile UB

Cecha \ L.p.	1	2	3	4	5	6	7	8	9	10	11	12
Wiek	28	22	23	23	22	24	21	24	20	20	23	23
Płeć	M	M	M	M	M	M	K	M	K	K	K	K
Bezpieczeństwo podczas logow.	7	7	8	7	9	10	9	8	8	8	10	9
Bezpieczeństwo bankow. mob.	2	4	5	6	7	8	8	9	9	9	10	10
Oprawa graficzna	8	7	7	5	9	10	8	8	8	9	8	7
Zrozumiałość	9	9	8	6	8	8	8	8	10	9	7	7
Łatwość obsługi	8	8	6	5	9	9	8	9	8	8	6	7
Komfort	7	7	5	6	10	10	7	9	9	8	9	6
Liczba funkcji	6	7	8	5	9	8	9	9	8	8	10	10

Wśród odpowiedzi dotyczących mBanku mężczyźni stanowili 53% ankietowanych, natomiast kobiety odpowiednio 47%. Respondenci to w większości osoby w wieku od 21 do 24 lat. Wyniki badań ankietowych dotyczące interfejsu aplikacji mobilnej mBanku zostały zaprezentowane w tab. 4.

Tabela 4. Wyniki badań ankietowych dotyczące aplikacji mBanku

Cecha \ L.p.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Wiek	24	21	21	23	23	24	21	24	19	23	29	22	25	24	23
Płeć	M	K	M	M	K	M	K	M	K	M	K	K	K	M	M
Bezp. podczas log.	3	7	7	6	8	9	8	6	9	8	8	8	9	8	8
Bezp. banków. mob.	1	4	6	6	7	8	8	8	9	9	9	9	10	10	10
Oprawa graficzna	8	7	8	9	7	10	8	10	7	6	9	10	8	10	7
Zrozumiałość	8	6	6	6	3	9	6	9	9	4	6	10	8	7	7
Łatwość obsługi	4	8	7	6	6	9	7	9	8	7	8	10	9	7	7
Komfort	4	8	7	6	6	10	8	9	8	8	8	10	9	9	8
Liczba funkcji	5	7	7	7	7	9	9	9	6	8	8	8	9	8	9

3.2. Testy korytarzowe

W trakcie testów korytarzowych użytkownicy wypełnili 11 kwestionariuszy i wyrazili swoją opinię na temat funkcjonalności oraz cech każdego z badanych interfejsów mobilnych aplikacji bankowych (tab. 5).

Tabela 5. Wyniki testów korytarzowych

mBank											
Sprawdzanie stanu konta	10	10	10	10	7	10	10	10	8	9	10
Wykonanie przelewu	8	8	10	10	8	8	10	8	8	9	10
Zmiana kodu PIN	7	8	10	9	10	6	9	8	5	8	8
Utworzenie lokaty	10	7	10	9	9	5	7	8	3	8	9
Zlokalizowanie bankomatu	3	9	4	1	5	3	9	4	4	4	4
Ocena responsywności	9	9	10	8	9	8	9	8	8	7	9
Ocena wizualna	8	6	6	7	7	7	8	8	9	7	9
L.p.	1	2	3	4	5	6	7	8	9	10	**11**
Sprawdzanie stanu konta	10	10	10	10	10	10	10	10	10	10	10
Wykonanie przelewu	8	8	10	5	10	9	10	8	9	9	5
Zmiana kodu PIN	8	7	10	9	10	8	10	8	9	9	3
Utworzenie lokaty	9	8	10	3	10	7	9	7	10	9	4
Zlokalizowanie bankomatu	9	7	10	10	10	7	9	9	10	9	9
Ocena responsywności	9	7	10	2	10	6	9	8	7	8	2
Ocena wizualna	9	5	8	5	10	6	10	8	9	8	4
										T-Mobile UB	

3.3. Badania eksperckie

W badaniach eksperckich, przeprowadzonych na grupie czterech doświadczonych użytkowników korzystających bardzo często z aplikacji oraz posiadających aplikację od co najmniej trzech miesięcy, uzyskano rezultaty, które przedstawia tab. 6.

Tabela 6. Badania eksperckie

Funcjonalność/Cecha	mBank		T-Mobile	
Sprawdzanie stanu konta	10	10	10	10
Wykonanie przelewu	8	8	8	9
Zmiana kodu PIN	9	9	10	10
Utworzenie lokaty	9	10	10	8
Zlokalizowanie bankomatu	10	6	10	10
Ocena responsywności	10	8	10	7
Ocena wizualna	9	7	9	10
Bezpieczeństwo aplikacji	10	9	10	10
Dostępność aplikacji	10	8	10	6
Dodatkowe formy płatności	10	10	9	8

3.4. Eyetracking

Podczas badań eyetrackingowych przedstawione zostało pięć scenariuszy przypadków użycia aplikacji. Wszystkie scenariusze testowe zostały wykonane na tym samym urządzeniu: Samsung Galaxy S7. Każdy użytkownik miał do wykonania następujące czynności:

I. Sprawdzenie stanu konta.
II. Wykonanie przelewu do konkretnej osoby z książki adresowej.
III. Zmiana kodu PIN.
IV. Utworzenie lokaty.
V. Zlokalizowanie bankomatu w pobliżu.

Testy przeprowadzone zostały na grupie dziesięciu osób. Podczas badania osoby przeprowadzające eksperyment sprawdzały następujące czynniki:

- pomyślność realizacji danego scenariusza,
- czasy skupienia wzroku na wybranych punktach,
- czas wykonywania zadania.

Czasy wykonania scenariuszy przedstawia tab. 7.

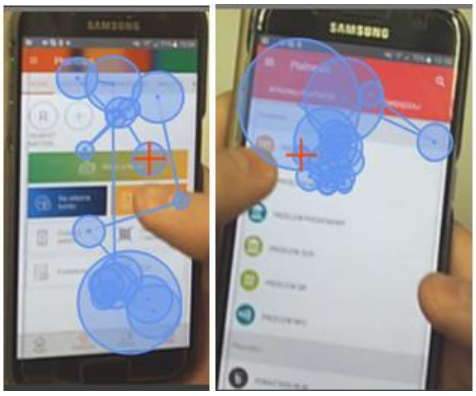
Tabela 7. Czasy wykonania poszczególnych scenariuszy [s]

Scen. / L.p.	I		II		III		IV		V	
Bank	M	T	M	T	M	T	M	T	M	T
1	22,07	9,68	27,02	31,47	34,45	27,81	22,7	13,21	159,8	10,81
2	4,46	1,65	28,41	32,86	25,85	34,01	18,26	17,26	52,96	7,68
3	2,63	2,17	22,25	48,15	33,67	26,07	21,78	47,42	54,57	6,55
4	3,16	1,91	28,77	69,4	22,91	24,21	12,89	13,88	92	5,77
5	14,27	3,22	46,65	17	23,99	21,81	18,58	13,86	14,85	6,03
6	65,4	4,07	30,62	36,89	123,94	49,58	107,68	33,68	161,17	4,78
7	6,98	2,37	33,98	52,08	25,6	25,57	16,17	31,66	43,16	3,54
8	19,02	4,59	29,64	24,16	24,53	20,28	14,34	9,55	32,05	2,83
9	2,9	1,33	20,55	43,38	27,27	73,01	14,46	22,83	35,17	6,94
10	2,91	2,75	37,5	24,71	28,06	25,78	15,58	15,77	6,88	6,48
t	14,38	3,37	30,54	38,01	37,03	32,81	26,24	21,91	65,26	6,14

Oznaczenia:

M – mBank,
T – T-Mobile Usługi Bankowe,
t – średni czas wykonania danego scenariusza.

Przykładowe ścieżki wzroku przedstawia rys. 1.



Rysunek 1. Ścieżki wzroku dla sprawdzania salda konta mBank i sprawdzania salda konta T-Mobile UB

4. Analiza danych i interpretacja

4.1. Badania ankietowe

W tab. 8 zostały przedstawione wyniki testu Shapiro-Wilka dla badanych cech aplikacji T-Mobile Usług Bankowych. Współczynnik SW oznacza wartość odczytaną z tablic Shapiro-Wilka dla liczności próby n=12 oraz poziomu istotności 0,05 , który został dopasowany do liczebności prób.

Tabela 8. Test SW dla ankiet dot. T-Mobile UB

SW= *0,8590*	Bezp. podczas logow.	Bezp. bankow. mob.	Ocena wizual.	Zrozu-miałość	Łatwość obsługi	Komfort	Liczba funkcji
W	0,8914	0,9051	0,9195	0,9391	0,8672	0,9298	0,9199
Norm.?	tak	tak	tak	tak	tak	tak	tak

Tab. 9 przedstawia wynik przeprowadzonego testu Shapiro-Wilka dla badanych cech aplikacji mBanku. SW oznacza wartość odczytaną z tablic dla liczności próby n=15 oraz poziomu istotności 0,05.

Tabela 9. Test SW dla ankiet dotyczących mBanku

SW= 0,8809	Bezp. podczas logow.	Bezp. bankow. mobilnej	Oprawa graficzna	Zrozu-miałość	Łatw. obsługi	Komfort	Liczba funkcji
W	0,7872	0,8407	0,8943	0,88099	0,9437	0,9002	0,8906
Normalny?	nie	nie	tak	tak	tak	tak	tak

W tab. 10 zaprezentowane są wyniki testu t-Studenta badanych aplikacji, dla cech, które w obydwu przypadkach przeszły pomyślnie test Shapiro-Wilka.

Tabela 10. Test t-Studenta wyników badań ankietowych

t=1,71	Oprawa graficzna	Zrozumia-łość	Łatwość obsługi	Komfort	Liczba funkcji
T	0,86	-1,83	-0,21	0,19	-0,67

Z obliczonych statystyk testu t-Studenta (T), przy odczytanej z tablic wartości t=1,71 stwierdza się, że na poziomie istotności 0,05 aplikacja T-Mobile Usług Bankowych jest bardziej zrozumiała od aplikacji mobilnej mBanku oraz, że nie ma istotnych różnic pomiędzy aplikacjami w pozostałych czterech kategoriach.
Dla cech nieposiadających rozkładu normalnego policzone zostały statystyki opisowe oraz odchylenia standardowe (tab. 11).

Tabela 11 Statystyki opisowe badań ankietowych

Zadanie	mBank				T-Mobile UB			
	śred.	med.	mod.	odch.	śred.	med.	mod.	odch.
Bezp. podczas logow.	7,47	8	8	1,500	8,33	8	8	1,028
Bezp. bank. mobilnej	7,61	8	9	2,417	7,25	8	8	2,419

Z otrzymanych w tab. 11 rezultatów wynika, iż aplikacja mobilna T-Mobile UB została lepiej oceniona pod kątem bezpieczeństwa w trakcie logowania uzyskując wyższą średnią arytmetyczną ocen oraz posiadając niższe odchylenie standardowe wyników. Z drugiej strony klienci mBanku znacznie częściej wyrażali opinię, iż uważają bankowość mobilną za bezpieczną formę zarządzania kontem bankowym. Aplikacja mBanku w tej kategorii uzyskała wyższą średnią arytmetyczną ocen od aplikacji mBanku przy niemal identycznym rozrzucie wartości.

Na podstawie uzyskanych w badaniu ankietowym wyników sporządzono zestawienie, które przedstawia tab. 12.

Tabela 12. Oceny cząstkowe w badaniach ankietowych

	Punkty cząstkowe	
Kryterium\Bank	mBank	T-Mobile UB
K1 Bezpieczeństwo	7	8
K4 Zrozumiałość	7	8
K5 Komfort pracy	8	8
K6 Prostota	8	8
K7 Liczba sposobów dokonywania płatności	8	8
K8 Ocena wizualna	8	8

4.2. Testy korytarzowe

W tab. 13 przedstawione zostały wyniki testu Shapiro-Wilka dla ocen wystawionych aplikacjom mobilnym mBank i T-Mobil UB pod kątem badanych cech.

Tabela 13. Test SW dla badań korytarzowych

SW=0,7920	mBank	T-Mobile UB
Sprawdzanie stanu konta	0,6153 X	0 X
Wykonanie przelewu	0,6963 X	0,8153 ✓
Zmiana kodu PIN	0,9288 ✓	0,7747 X
Utworzenie lokaty	0,8829 ✓	0,8363 ✓
Zlokalizowanie bankomatu	0,8158 ✓	0,7720 X
Ocena responsywności	0,8924 ✓	0,8450 ✓
Ocena wizualna	0,9034 ✓	0,9017 ✓

Wyniki testu Shapiro-Wilka ukazały, że jedynie trzy z siedmiu pozycji ocenianych w testach korytarzowych ma rozkład zbliżony do normalnego i co za tym idzie może zostać poddana testowi t-Studenta, który ze względu na małą liczebność próby zostanie przeprowadzony dla poziomu istotności α=0,1 (tab. 14).

Tabela 14. Test t-Studenta wybranych pozycji testów korytarzowych

Zadanie/Ocena	Statystyka t
Utworzenie lokaty	-0,09
Ocena responsywności	1,65
Ocena wizualna	0

Odczytując z tablicy t-Studenta wartość $t_{\alpha}^{n_1+n_2-2} = 1{,}325$ można stwierdzić, że na poziomie ufności rzędu 90% aplikacja mBanku jest bardziej responsywna od aplikacji mobilnej T-Mobile UB oraz, że utworzenie lokaty w obu aplikacjach ma taki sam stopień trudności. Aplikacje mobilne pod kątem wizualnym są na tym samym poziomie.

Tabela 15. Statystyki opisowe cech niepodlegających rozkładowi Gaussa

Zadanie	mBank				T-Mobile UB			
	śred.	med.	mod.	odch.	śred.	med.	mod.	odch.
Sprawdzanie stanu konta	9,45	10	10	0,987	10	10	10	0
Wykonanie przelewu	8,82	8	8	0,936	8,27	9	8	1,710
Zmiana kodu PIN	8	8	8	1,477	8,27	9	8	1,911
Odnalezienie bankomatu	4,55	4	4	2,310	7,82	9	9	1,044

Analizując miary położenia rozkładów oraz odchylenia standardowe (tab. 15) można stwierdzić, iż:

- sprawdzenie stanu konta w przypadku interfejsu aplikacji T-Mobile UB jest prostszym zadaniem, ponieważ 2 z 3 miar jest dla tej aplikacji większych, jedna równa, a odchylenie standardowe jest zerowe;
- wykonanie przelewu prostsze jest w aplikacji mBanku, ponieważ aplikacja T-Mobile UB pomimo większej mediany posiada niższą średnią arytmetyczną ocen oraz cechuje się dużym rozrzutem wartości ocen;
- zmiana kodu PIN w przypadku obu interfejsów cechuje się podobnym stopniem skomplikowania, ponieważ aplikacja T-Mobile UB pomimo lepszego rezultatu osiągniętego dla miar położenia cechuje się znacznym odchyleniem standardowym, co potwierdzają wszystkie z uwzględnianych miar;
- lokalizacja najbliższego bankomatu jest prostszym zadaniem w przypadku T-Mobile UB.

Oceny cząstk. testów korytarzowych przedstawia tab. 16.

Tabela 16. Oceny cząstkowe w badaniach korytarzowych

	Punkty cząstkowe	
Kryterium\Bank	mBank	T-Mobile UB
K2 Średni czas realizacji danej czynności	9	8
K4 Zrozumiałość	7	8
K6 Prostota	7	8
K8 Ocena wizualna	7	7

4.3. Badania eskperckie

Na podstawie opinii wyrażonych przez ekspertów stwierdza się, że dla zaawansowanych użytkowników:

- sprawdzenie salda konta i wykonanie przelewu jest łatwiejsze w przypadku aplikacji T-Mobile UB,
- poziom trudności odnalezienia opcji „zmień PIN" w oknie ustawień oraz zakładki lokaty w oknie finanse był zbliżony w przypadku obu aplikacji,
- znalezienie bankomatu było znacznie łatwiejsze i intuicyjne w przypadku aplikacji T-Mobile UB.

Punkty cząstkowe w badaniach eksperckich zaprezentowane zostały w tab. 17.

Tabela 17. Oceny cząstkowe w badaniach eksperckich

	Punkty cząstkowe	
Kryterium\Bank	mBank	T-Mobile UB
K1 Bezpieczeństwo	10	10
K3 Dostępność	10	8
K5 Komfort pracy	9	9
K7 Liczba sposobów dokonywania płatności	10	9
K8 Ocena wizualna	8	10

4.4. Badania eyetrackingowe

Poddając analizie czasy wykonania poszczególnych scenariuszy można stwierdzić, że:

- sprawdzenie stanu konta zajęło średnio ponad 4 razy więcej czasu w przypadku aplikacji mBank,
- wykonanie przelewu zajęło podobną ilość czasu w przypadku obu aplikacji średnio 24,46% więcej czasu w przypadku aplikacji T-Mobile UB,
- zmiana kodu PIN zajęła podobną ilość czasu w przypadku obu aplikacji,
- utworzenie lokaty zajęło podobną ilość czasu w przypadku obu aplikacji średnio 19,76% więcej czasu w przypadku aplikacji mBank,
- zlokalizowanie bankomatu zajęło średnio ponad 10 razy więcej czasu w aplikacji mobilnej mBank.

Podsumowując, aplikacja mBank wykazała się lepiej w przypadku czasu wykonania przelewu, natomiast aplikacja T-Mobile UB była łatwiejsza w przypadku zmiany

kodu PIN i utworzenia lokaty oraz łatwiejsza w przypadku sprawdzenia stanu konta i zlokalizowania bankomatu.

Analizując ścieżki wzroku poszczególnych scenariuszy dla grupy 10 osób można stwierdzić, że:

- sprawdzenie salda konta oraz wykonanie przelewu jest łatwiejsze i bardziej intuicyjne w przypadku aplikacji T-Mobile UB,
- poziom trudności odnalezienia opcji „zmień PIN" w oknie ustawień oraz zakładki lokaty w oknie finanse był zbliżony w przypadku obu aplikacji,
- znalezienie bankomatu było prostsze i bardziej intuicyjne w przypadku aplikacji T-Mobile UB.

Analiza map cieplnych oraz widm dała takie same wnioski jak w przypadku analizy ścieżek wzroku.

Oceny cząstkowe w badaniu okulograficznym zaprezentowane zostały w tab. 18.

Tabela 18. Oceny cząstkowe w badaniach eyetrackingowych

	Punkty cząstkowe	
Kryterium\Bank	mBank	T-Mobile UB
K4 Zrozumiałość	8	7
K6 Prostota	6	9

5. Ocena końcowa i wnioski

Na podstawie analizy przeprowadzonej analizy danych oraz w oparciu o wzór (1) sporządzono następujące zestawienie bankowych aplikacji mobilnych na tle sporządzonych kryteriów porównawczych (tab. 1).

Tabela 19. Punktacja końcowa interfejsów

Kryterium	Ocena	
	mBank	T-Mobile UB
Bezpieczeństwo	20,4	21,6
Śr. czas realizacji zadania	17,1	15,2
Dostępność	15,0	12,0
Zrozumiałość	9,5	10,0
Komfort pracy	9,3	9,3
Prostota	6,3	7,5
Liczba spos. dok. płatności	5,4	5,1
Ocena wizualna	2,25	2,25
Wynik sumaryczny	85	83

Ostateczny wynik ukazuje, że aplikacja mBank, która uzyskała 85/100 punktów jest lepsza od aplikacja T-Mobile Usługi Bankowe, która zdobyła 83/100 punktów. Wyniki dla obu aplikacji są jednak bardzo zbliżone. Różnica wynosi jedynie dwa punkty.

Literatura

[1] A. Memon, "*User Interface Design: Designing effective interfaces for software systems",* College Park: University of Maryland, 2011.

[2] T. Mendel, "*The Elements of User Interface Design",* Nowy Jork: John Wiley & Sons, 1997.

[3] K. Ormaniec, „*Najlepsze aplikacje bankowe w Polsce – ranking Komórkomat.pl*", Komórkomat.pl, 12 08 2016. [Online]. Available: https://komorkomat.pl/ najlepsze-aplikacje-bankowe/. [Data uzyskania dostępu: 12 04 2017].

[4] A. Marcus, "*Cross-Cultural User-Interface Design for Work, Home, Play and on the Way",* Santa Fe: ACM SIGGRAPH ASIA, 2010.

[5] M. Miłosz, „*Ergonomia Systemów Informatycznych",* Lublin: Politechnika Lubelska, 2014.

[6] B. Szymańska, „*najlepszekonto.pl*", [Online]. Available: http://www.najlepszekonto.pl/bezpieczenstwo-bankowosci-mobilnej. [Data uzyskania dostępu: 20 04 2017].

[7] M. Mikowska, „*BANK.JEST.MOBI 2016*", jestem.mobi, Katowice, 2016.

[8] K. Śladkowska, „*5 problemów polskiej bankowości mobilnej*", bankier.pl, 17 05 2016. [Online]. Available: http://www.bankier.pl/wiadomosc/5-problemow-polskiej-bankowosci-mobilnej-7390809.html. [Data uzyskania dostępu: 22 04 2017].

[9] M. Jun i S. Palacios, "*Examining the key dimensions of mobile banking service quality: an exploratory study"*, Las Cruces: International Journal of Bank Marketing, 2016.

[10] L. Fabisiak i P. Ziemba, „*Wybrane metody analizy wielokryterialnej w ocenie użyteczności serwisów internetowych"*, Szczecin: Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, 2011.

[11] A. Smoliński, „*Klasyfikacja i porównanie metod klasycznej optymalizacji wielokryterialnej"*, Szczecin: Zachodniopomorski Uniwersytet Technologiczny w Szczecinie, 2014.

JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 167-173
Wysłane: 2017-06-21
Przyjęte: 2017-06-28

Analiza wydajnościowa silnika Unity3D w kontekście aplikacji uruchamianych w przeglądarkach internetowych

Przemysław Berdak*, Małgorzata Plechawska-Wójcik
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Tematem artykułu są badania wydajnościowe silnika Unity3D do tworzenia aplikacji, które są uruchamiane z wykorzystaniem przeglądarki internetowej oraz technologii WebGL. W pracy wykonany został przegląd literaturowy z zakresu tworzenia aplikacji multimedialnych uruchamianych w przeglądarkach internetowych, silników graficznych, czy technologii WebGL. Dodatkowo omówione zostały zagadnienia techniczne związane z konwertowaniem aplikacji tworzonych w silniku Unity3D do WebGL. Realizacja badań oparta jest o symulację komputerową z użyciem pięciu scen w środowisku Unity3D, które zawierają główne możliwości tego silnika. Na końcu zaprezentowano wyniki owych badań oraz przedstawiono wnioski wyciągnięte na ich podstawie.

Słowa kluczowe: Unity3D; WebGL; Przeglądarki Internetowe; Silniki Graficzne

Autor do korespondencji *.

Adres e-mail: przemyslaw.berdak@gmail.com

Performance analysis of Unity3D engine in the context of applications run in web browsers

Przemysław Berdak*, Małgorzata Plechawska-Wójcik
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The aim of the work is the performance analysis of the Unity3D engine in the contex of applications running with a web browser and WebGL technology.. The paper presents literature review was carried out in the scope of creating multimedia applications run in web browsers, graphics engines and WebGL technology . Additionally, technical issues related with converting Unity3D application to WebGL technology are discussed. The research are performed based on computer simulations with five scenes created with Unity3D environment, which include the main capabilities of this engine. Finally, it was presented the results of these studies and the conclusions.

Keywords: Unity3D; WebGL; Web Browsers; Graphics Engines

Corresponding author *.

E-mail address: przemyslaw.berdak@gmail.com

1. Wstęp

Sieć Internet oraz przeglądarki internetowe stają się coraz popularniejszą platformą aplikacyjną, której można użyć do publikowania aplikacji multimedialnych, takich jak na przykład gry wideo. Przez lata istniały różne metody tworzenia takich aplikacji, lecz z różnymi możliwościami graficznymi czy fizycznymi oraz z róż nym stopniem skomplikowania. Od roku 2009 takie aplikacje można tworzyć z wykorzystaniem technologii WebGL, która rozszerza język JavaScript o elementy API do tworzenia grafiki bazującej na OpenGL ES. Taki podzbiór może zostać użyty na przykład przez bibliotekę graficzną, lecz może mieć ona zbyt mało możliwości, aby stworzyć nowoczesną, bogatą w wiele funkcji aplikację multimedialną. Rozwiązaniem może być użycie silnika graficznego, który oferuje bardzo bogate możliwości rozwoju, nawet bardzo skomplikowanych aplikacji, lecz do niedawna mogły być one uruchamiane tylko desktopowo. Od maja 2015 roku silnik Unity3D oferuje możliwość przekonwertowania nawet bardzo skomplikowanych aplikacji do technologii WebGL, dzięki czemu mogą być one uruchomione z wykorzystaniem przeglądarki internetowej. Takie aplikacje, według twórców Unity3D, jakościowo stoją na bardzo zbliżonym, o ile nie na takim samym poziomie, co aplikacje natywne generowane do kodu odpowiadającego danej platformie.

Z faktu, że takie rozwiązanie jest stosunkowo nowe pojawia się konieczność zbadania jego wydajności, aby określić możliwości poprawnej pracy pod różnymi przeglądarkami internetowymi. Z tego też powodu głównym celem pracy dyplomowej, której dotyczy niniejszy artykuł jest zbadanie wydajności silnika Unity3D w powyższym zakresie i wyciągnięcie wniosków dotyczących możliwości jego stosowalności w kontekście przeglądarek internetowych. Realizacja tak omówionego celu polegała na wygenerowaniu pięciu scen, które były badane w powyższym kontekście.

2. Badania literaturowe

Badania literaturowe rozpoczęto od artykułu omawiającego przegląd głównych sposobów renderowania grafiki 3D poprzez przeglądarkę internetową. Takim artykułem był *3D graphics on the web: A survey* autorstwa *Aluna Evansa, Maco Romeo, Arasha Behrehmanda, Javiego Agenjo i Josepa Blata.*[1] Autorzy przywołali tutaj podział

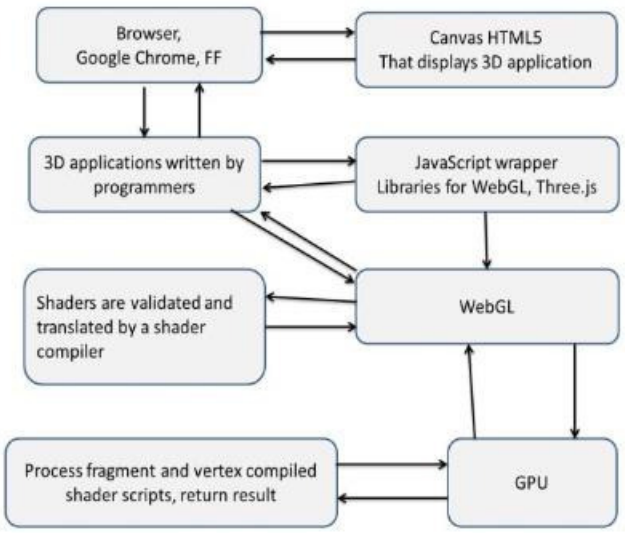
sposobu tworzenia grafiki wyświetlanej przez przeglądarkę internetową na imperatywny i deklaratywny, który może być stosowany do grafiki 2D oraz 3D. Grafika deklaratywna i imperatywna może dotyczyć zarówno grafiki dwuwymiarowej jak i trójwymiarowej. W przypadku grafiki dwuwymiarowej podejściem deklaratywnym może być tworzenie grafiki wektorowej, natomiast podejściem imperaktywnym rysowanie na elemencie *Canvas*. W opisie imperatywnego sposobu renderowania grafiki trójwymiarowej autorzy tego artykułu przywołują przykład pluginu Adobe Flash, który pozwala na uruchomienie zawartości multimedialnej wewnątrz strony internetowej. Poza technologią Adobe Flash przywołano tutaj inne sposoby tworzenia aplikacji uruchamianych w przeglądarkach internetowych, takich jak Microsoft Silverlight, czy Google O3D. Publikacja kończy się nawiązaniem do renderowania grafiki z wykorzystaniem elementu Canvas, takie podejście może być, według autorów, traktowane jako prekursor technologii WebGL.

Uzupełnieniem powyższego może być artykuł *On the Development of Browser Games - Technologies of an Emerging Genre* Juha-Mattiego Vanhatupa [2]. Tutaj też przedstawiono przegląd poszczególnych technologii umożliwiających uruchomienia aplikacji z użyciem przeglądarki internetowej. Technologie Adobe Flash, czy Microsoft Silverlight zostały przez autora uzupełnione przez Shockwave, czy plugin Unity Web Player.

Takie technologie bardzo często nie wykorzystywały możliwości silników graficznych, do czego wyjątkiem jest wspomniany plugin Unity Web Player, dlatego też w ramach badań literaturowych przyjrzano się bliżej zagadnieniu związanym z silnikami graficznymi. Ciekawą pracą naukową mogącą rzucić więcej światła na dane zagadnienie jest *Graphics Engine Technology Research and Implementation* autorstwa Liu Yijun, Chen Wenbin i He Xiaoman [3]. Publikacja ta została opublikowana jako wynik prac związanych z opracowaniem konceptu i implementacji silnika graficznego. Poruszona jest tutaj kwestia istotności realistyczności grafiki w wielu aplikacjach, które mają za zadanie tworzyć animacje filmowe, rozszerzoną rzeczywistość, gry wideo, czy obiekty architektoniczne. Autorzy publikacji zauważają, że silnik graficzny składa się z wielu elementów i nie służy tylko do renderowania grafiki. Typowy silnik graficzny składa się również z silnika fizyki, systemu animacji, czy bibliotek modeli, obrazów oraz dźwięków. Jego kwintesencją jest jednak rysowanie obrazu dwuwymiarowego na podstawie sceny trójwymiarowej, dlatego też ten element powinien być najlepiej dopracowany.

Dobry silnik graficzny musi być też dobrze zorganizowany, aby tworzone aplikacje wyglądały i działały możliwie jak najlepiej. W publikacji publikacji *A Survey of Frameworks and Game Engines for Serious Game Development,* której autorami są Brent Cowan oraz Bill Kapralos [4] przedstawiono kilka cech dobrego silnika graficznego, do których zaliczają się między innymi możliwość skryptowania obiektów, renderowania grafiki, animacje oraz właściwości fizyczne, a także edytory poszczególnych tych cech.

Dzięki publikacji *The The Web as an Application Platform: The Saga Continues* autorstwa Antero Taivalsaariego i Tommiego Mikkonena [5] można przejść z tematu silników graficznych do tematu użycia sieci Internet oraz przeglądarek internetowych jako platformy, która może zostać użyta do uruchamiania aplikacji. Publikacja skupia się na możliwości wykorzystania HTML5 i WebGL jako narzędzi eliminujących ograniczenia sieci i przekształcających sieć WWW w platformę aplikacyjną, gdyż według autorów w przyszłości większość oprogramowania dostarczana będzie właśnie przez sieć Internet.



Rys. 1. Stos WebGLa według Rovshena Nazarova i John Galletly [6]

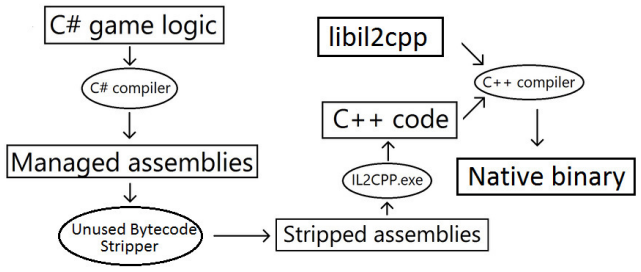
Możliwość wykorzystania technologii WebGL i HTML5 do tworzenia aplikacji multimedialnych w swojej pracy *Native browser support for 3D rendering and physic using WebGL, HTML5 and Javascript* poruszają też Rovshen Nazarow oraz John Galletly [6], którzy zaprezentowali sposób wyświetlania grafiki z użyciem tych technologii. W pracy tej zostało dokładniej omówione zagadnienie WebGL, natomiast autorzy przedstawili również stos tej technologii widoczny na rysunku numer 1.

3. Opis obiektu badań

Silnik Unity3D jest jednym z najpopularniejszych silników graficznych przeznaczonych do tworzenia trójwymiarowych gier wideo oraz innych materiałów. Silnik ten dostarcza szereg komponentów dla twórców gier, aby mogli oni zamodelować to co chcą z możliwie największą dokładnością, a sam proces był możliwie najłatwiejszy. Silnik ten dostarcza praktycznie wszystkie elementy konieczne do tworzenia efektownych aplikacji, co czyni go jednym

z najpopularniejszych silników na świecie. Poza typowymi elementami, takimi jak możliwość tworzenia scen, grafiki, shaderów, czy animacji silnik Unity dostarcza możliwość tworzenia skryptów, które można napisać w języku C#, bądź JavaScript [7].

Wszystkie elementy aplikacji są jednak tłumaczone na natywny kod obsługiwanej platformy, czyli jednej z dwudziestu dwóch, które obsługuje Unity3D. Aby kod aplikacji został przetłumaczony na jedną z tych platform musi istnieć mechanizm, który kod aplikacji przetłumaczy na odpowiedni język, najczęściej C/C++. Takim mechanizmem jest kompilator IL2CPP, który służy do przetwarzania kodu napisanego w języku C# do języka właśnie C/C++, proces ten przedstawiono na rysunku numer 2. [8]



Rys. 2. Diagram budowy kodu C++ na podstawie C# z wykorzystaniem technologii IL2CPP [8]

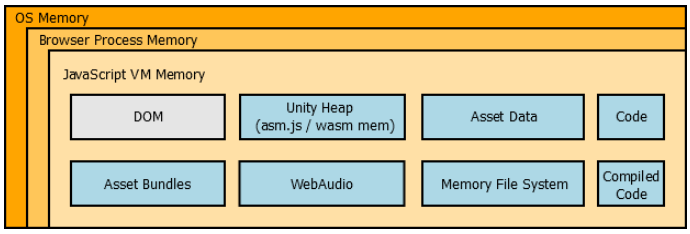
Tak powstały kod jest dodatkowo kompilowany do języka asm.js, który jest podzbiorem języka JavaScript. Został on opracowany aby wydajnościowo zbliżyć język JavaScript do poziomu kodu natywnego i według autorów kod natywny jest tylko 1.5 razy szybszy. Aby kod powstały powyżej mógł zostać skompilowany do kodu asm.js wykorzystywany jest kompilator Emscripten oraz kod LLVM, a schemat postępowania w przypadku takiego kodu jest widoczny na rysunku numer 3. [9]

Rys. 3. Schemat kompilacji kodu C/C++ do JavaScript [9]

Tak powstały kod w języku JavaScript wykorzystuje następnie technologię WebGL, która dostarcza API do interfejsu wyświetlania grafiki bazującego na OpenGL ES, który jest wykorzystywany na elemencie Canvas HTML5. [9]

Jako, że WebGL bazuje na JavaScripcie, jego wydajność jest bezpośrednio powiązana z prędkością tego języka i choć jeszcze kilka lat temu jego wydajność była niezadowalająca, dzisiaj może być używany do tworzenia wydajnych aplikacji renderujących grafikę trójwymiarową. Z powodu swojej popularności standard ten jest obecnie wspierany w różnym stopniu przez niemal wszystkie przeglądarki internetowe, zarówno te desktopowe, jak i mobilne. Na rysunku numer 4 widoczne są wszystkie komponenty pamięci, które powstają dla aplikacji WebGL wygenerowanej za pomocą Unity3D. [9]

Jednym z większych problemów podczas tworzenia aplikacji tworzonych z użyciem silnika Unity3D, które mają zostać uruchomione w przeglądarce internetowej jest problem z niewystarczającą ilością pamięci. Użytkownicy końcowi mogą uruchomić aplikację na różnych przeglądarkach, na różnych urządzeniach oraz na różnych systemach operacyjnych. W związku z tym konieczne jest nie tylko odpowiednie zarządzanie pamięcią, ale także odpowiednia kompresja zasobów aplikacji [10].



Rys. 4. Rozmieszczenie poszczególnych elementów aplikacji Unity WebGL w przeglądarce internetowej [10]

Niestety problemy z pamięcią nie są jedynymi problemami aplikacji tworzonymi z użyciem silnika Unity3D i konwertowanych do WebGL. Spośród nieobsługiwanych przez WebGL elementów systemu Unity, który najbardziej może zaważyć na efektywności takich aplikacji jest brak wielowątkowości. Spowodowane jest to brakiem wątków w języku JavaScript, do którego konwertowane są zasoby Unity. Dodatkowo mogą wystąpić pewne limity związane z wyświetlaniem grafiki spowodowane limitami w OpenGL ES 2.0, a także limity związane z bardziej zaawansowanym audio. Z ograniczeń ściśle programistycznych nie ma możliwości użycia programowania refleksyjnego.[9]

Tak tworzona i uruchamiana aplikacja jest poddawana badaniom w pracy dyplomowej, na bazie której opracowana została dana publikacja.

4. Opis metodyki badawczej

Badanie zostało przeprowadzone z wykorzystaniem eksperymentu polegającego na wykorzystaniu symulacji komputerowej. Metodyka badawcza zakłada zdefiniowanie pięciu scenariuszy badawczych, które podlegają symulacji komputerowej. Wyniki takiej symulacji są porównywane z symulacją przeprowadzoną dla aplikacji wygenerowanych do kodu natywnego oraz jako aplikacja mobilna.

Badanymi elementami silnika Unity3D są:

- Animacje
- Grafika i shadery
- System cząstek
- Trójwymiarowa fizyka obiektów
- Generowanie i usuwanie obiektów

Aby przeprowadzić taką analizę należy wygenerować kilka scen, które będą zawierać elementy dotyczące każdego z powyższych kryteriów. Każda tak wygenerowana scena będzie oddzielnym scenariuszem testowym. Takie sceny powinny być uruchamiane oddzielnie na platformie testowej.

Dodatkowo badanie jest przeprowadzane pod różnymi

przeglądarkami internetowymi.

Wśród badanych przeglądarek wyróżniono:

- Mozilla Firefox 52
- Google Chrome 58
- Microsoft Edge 38
- Opera 45

Aby porównać wydajność badanej technologii badania są przeprowadzane również dla scen wygenerowanych dla aplikacji natywnej uruchamianej na laptopie oraz dla natywnej aplikacji natywnej uruchamianej na telefonie komórkowym. Platformą testową jest laptop z procesorem Intel Core i3 2370M 2,4GHz, 4 GB RAM DDR3 i kartą graficzną AMD Radeon HD 7670M + Intel HD Graphics 3000, natomiast użytym sysemem operacyjnym był Windows 10. Do badań nad aplikacją natywną wykorzystano smartfon LG G3 z systemem operacyjnym Google Android 6.0, procesorem czterordzeniowym Qualcomm Snapdragon 801 2.50 GHz, układem graficznym Adreno 330 i GB pamięci RAM.

Metodyka badawcza wymaga też zdefiniowania kryteriów porównawczych, które opiszą przeprowadzony eksperyment. Wśród tych zdefiniowanych znajdują się:

- Czas całkowitego załadowania aplikacji (sekundy)
- Średnia liczba klatek na sekundę (klatki na sekundę)
- Zajętość pamięci w systemie operacyjnym przeglądarki z aplikacją, bądź samej aplikacji w przypadku aplikacji natywnych (megabajty)
- Procent zużycia procesora (procent)

Metodyka badawcza zakłada także opracowanie wyników eksperymentu z wykorzystaniem zagadnienia analizy wielokryterialnej. Do analizy została wybrana Metoda Sumy Ważonej, natomiast normalizacja wyników została przeprowadzona Metodą Unitaryzacji Zerowej.

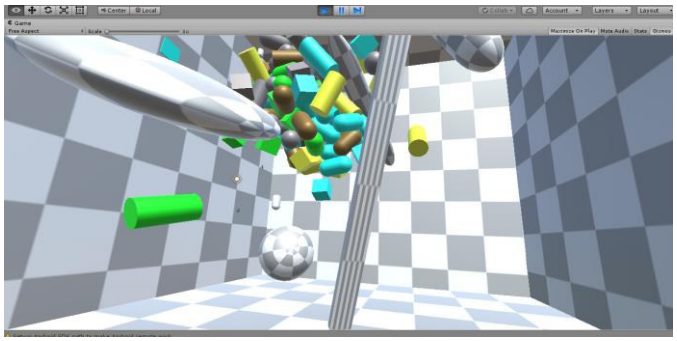
5. Model symulacji

Ze względu na dobrane kryteria oraz zdefiniowane sceny na potrzeby badań użyto pięciu różnych aplikacji, z których każda była odpowiedzialna za zbadanie innego komponentu silnika Unity3D w kontekście aplikacji uruchamianych w przeglądarkach internetowych.

Przechodząc do poszczególnych scen pierwszą z nich jest scena bazująca na animacjach. Aby wygenerować taką scenę należy wykorzystać kilka modeli, które wykonują jak najwięcej ruchów oraz animacji. Do sceny dodane zostało pięć głównych modeli z systemem animacji, a wszystkie te modele zostały następnie skopiowane pewną liczbę razy tak, aby na scenie było możliwie jak najwięcej animowanych obiektów. Do sceny dodana została trawa, na którą działają silne podmuchy wiatru, aby dołożyć do sceny jeszcze więcej animacji. Gotowa scena jest widoczna na rysunku piątym.

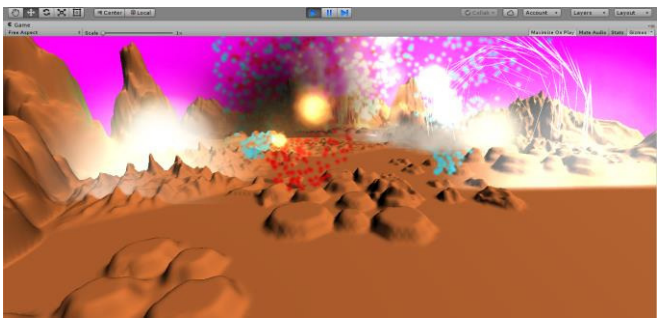
Rys. 5. Scena bazująca na animacjach

Drugą ze scen jaka zostanie przygotowana jest scena, której zadaniem jest zbadanie właściwości fizyki trójwymiarowej w silniku Unity3D. Dobrym pomysłem na zbudowanie takiej sceny jest utworzenie wielu prostych komponentów, a następnie dodaniu im różnych właściwości fizyki trójwymiarowej by w końcu wszystkie te zderzały się ze sobą. Wszystkie elementy znajdują się w kostce sześciennej, natomiast w czasie działania aplikacji zderzają się ze sobą. Wszystkie kolorowe elementy mają tutaj inne właściwości fizyczne, Scena jest widoczna na rysunku numer 6.



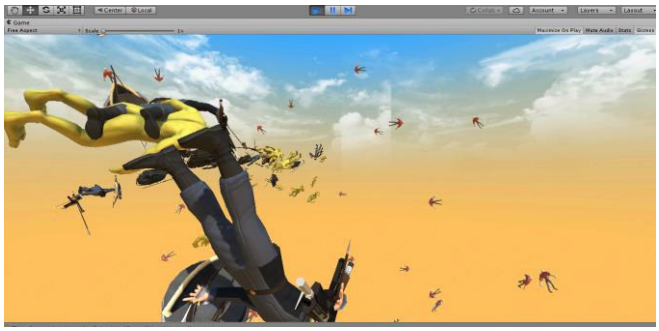
Rys. 6. Scena bazująca na fizyce trójwymiarowej

Trzecią sceną była scena, której zadaniem było wykonanie badań z pod kątem wykorzystania systemu cząsteczek. Scena taka została zbudowana z wykorzystaniem standardowego terenu oraz głównie standardowego systemu cząsteczek. Na wygenerowany teren zostało nałożone możliwie jak najwięcej systemów cząsteczek. Jest ona widoczna na szóstym rysunku 7.



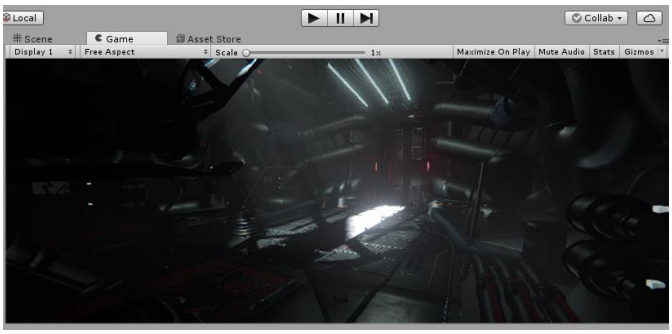
Rys. 7. Scena bazująca na systemie cząsteczek

Czwartą badaną sceną była scena badająca szybkość renderowania i usuwania obiektów gry. Do budowy sceny skorzystano głównie z tych samych elementów jak w przypadku sceny pierwszej oraz dodatkowego skryptu, którego zadaniem było zarządzanie generowaniem nowych i niszczeniem obiektów. Działanie tej sceny zostało zademonstrowane na rysunku 8.



Rys. 8. Scena bazująca na generowaniu i usuwaniu obiektów

Ostatnią z badanych scen jest scena, której zadaniem było zbadanie określenie wydajności aplikacji z zaawansowaną grafiką. W tym celu posłużono się sceną utworzoną przez *Unity Technologies* o nazwie *Adam Interior Scene*. Scena taka zawiera na tyle zaawansowane elementy graficzne, że idealnie nadaje się do badań. Została zaprezentowana na rysunku 9.



Rys. 9. Scena bazująca na grafice

6. Prezentacja rezultatów badań

Dla wszystkich omawianych scen zostały przeprowadzone badania. W tabeli pierwszej widoczne są wyniki badań dla sceny bazującej na animacjach w odniesieniu do zdefiniowanych kryteriów. Wyniki dla tej sceny zostały wybrane jako przykładowe, ponieważ są najbliżej średniej dla wyników ze wszystkich scen. Badania zostały przeprowadzone na systemie oraz sprzęcie przedstawionym w rozdziale czwartym. W tabeli widać, że aplikacje natywne osiągają większą liczbę klatek na sekundę od aplikacji przeglądarkowych oraz uruchamiają się zdecydowanie szybciej. Zużywają także mniej pamięci oraz mniej obciążają procesor.

Ocena możliwości wydajnościowych poszczególnych przeglądarek w kontekście aplikacji Unity WebGL uwzględniająca wszystkie wybrane kryteria wymaga przeprowadzenia analizy wielokryterialnej. Analiza taka może dać jednoznaczną odpowiedź na pytanie dotyczące analizy wydajnościowej silnika Unity3D w kontekście aplikacji uruchamianych w przeglądarkach internetowych

Tabela 1. Wyniki dla sceny bazującej na animacjach

		Czas załadowania [s]	Średnia liczba klatek na sekundę [fps]	Zajętość pamięci [mb]	Zużycie procesora [%]
Przeglądarki internetowe (Windows 10)	Mozilla Firefox 52	43.54	14	1828	26.7
	Google Chrome 58	45.26	13	2189	34.1
	Microsoft Edge 38	65.04	9	2305	33.3
	Opera 45	46.55	10	2158	22
Aplikacje natywne	Android	15.11	17	594	18
	Windows 10	10.48	21	835	14.8

W celu przeprowadzania analizy wielokryterialnej dane musiały zostać odpowiednio znormalizowane. Nastąpiło z użyciem Metody Unitaryzacji Zerowej. Metoda ta wymaga, aby każdy badany element został poddany normalizacji za pomocą stymulatora z życiem pierwszego wzoru poniżej albo destymulatora poprzez drugi z wymienionych niżej wzorów.

$$Z_{ij} = \frac{X_{ij} - \min(X_{ij})}{\max(X_{ij}) - \min(X_{ij})} \quad (1)$$

$$Z_{ij} = \frac{\max(X_{ij}) - X_{ij}}{\max(X_{ij}) - \min(X_{ij})} \quad (2)$$

Normalizacja została wykonana dla wszystkich otrzymanych wyników, natomiast tabela druga przedstawia wyniki normalizacji dla sceny bazującej na grafice.

Tabela 2. Normalizowane wyniki dla sceny bazującej na animacjach

		Czas załadowania	Średnia liczba klatek na	Zajętość pamięci	Zużycie procesora
Przeglądarki internetowe (Windows 10)	Mozilla Firefox 52	1	1	1	0.61
	Google Chrome 58	0.92	0,8	0.24	0
	Microsoft Edge 38	0	0	0	0.07
	Opera 45	0.86	0.2	0.31	1
Waga kryterium		0.3	0.4	0.2	0.1

Do przeprowadzenia analizy wielokryterialne konieczne jest ustalenie wag dla poszczególnych kryteriów. Przy ustalaniu wag uwzględniono, że dwoma kryteriami, które najbardziej wpływają na komfort korzystania z aplikacji jest liczba klatek na sekundę oraz czas załadowania aplikacji, dlatego też sumę ich wag ustalono na 0.7. Ważniejszym z tych dwóch jest kryterium określające liczbę klatek na sekundę, dlatego została mu przyznana waga 0.4, natomiast kryterium określającym czas załadowania 0.3. Przyglądając dwóm spośród pozostałych kryteriów zaobserwowano, że zbudowane aplikacje podczas działania procentowo wymagają większych zasobów pamięci niż procesora, dlatego też kryterium dotyczącym zajętości pamięci przydzielono wagę 02, natomiast kryterium związanym z zużyciem procesora przydzielona wagę 0.1.

Analiza wielokryterialna została wykonana Metodą Sumy Ważonej, gdzie były cztery główne kryteria zdefiniowane wcześniej oraz widoczne w tabeli pierwszej. Sama analiza została przeprowadzono z użyciem wzoru widocznego poniżej.

$$F(x) = \sum_{i=1}^{4} w_i f_i(x) \quad (3)$$

Wyniki analizy wielokryterialne są widoczne w tabeli trzeciej.

Tabela 3. Suma ważona jako wyniki analizy wielokryterialnej

	Animacje	Fizyka	System Cząsteczek	Gen/Usuw. obiektów	Grafika
Mozilla Firefox 52	0.961	0.3	0.571	0.751	0.108
Google Chrome 58	0.644	0.337	0.212	0.532	0.504
Microsoft Edge 38	0.007	0.506	0.477	0.035	0.6
Opera 45	0.5	0.4	0.4	0.742	0.57

Wyniki widoczne w tabeli trzeciej zostały dodatkowo uśrednione dla wszystkich kryteriów, aby zobaczyć która z przeglądarek najlepiej radzi sobie z uruchamianiem aplikacji Unity WebGL. Rezultaty badań na systemie operacyjnym Windows 10 są widoczne w czwartej tabeli. We wszystkich tabelkach wartości większe reprezentują lepszą wartość badanej cechy. Dodatkowo jako wyniki zaprezentowano wartości dla przeglądarek internetowych.

7. Dyskusja wyników i wnioski.

Otrzymane wyniki próbują odzwierciedlić jakość wykorzystania silnika Unity3D do budowy aplikacji, które mogą zostać uruchomione z użyciem przeglądarki internetowej. Porównując dane z przeglądarek internetowych można wskazać, że aplikacje natywne jednak lepiej radzą sobie z zadaną tematyką. Takie aplikacje z reguły osiągały lepsze wyniki od wszystkich przeglądarek internetowych. Wyniki aplikacji natywnych były lepsze we wszystkich badanych kryteriach, od czasu uruchomienia, poprzez liczbę klatek na sekundę aż do zasobności komponentów sprzętowych.

Tabela 4.Uśredniona suma ważona dla badanych przeglądarek

	Średnia suma ważona
Mozilla Firefox 52	0.5382
Google Chrome 58	0.4458
Microsoft Edge 38	0.325
Opera 45	0.5224

Żeby porównać wydajność poszczególnych przygotowanych scen oraz poszczególnych przeglądarek warto przejść do zestawienia zawierającego opracowanie wykonane z użyciem analizy wielokryterialnej. Można tutaj zauważyć, że przeglądarka Mozilla Firefox 52 najlepiej radzi sobie z animacjami poszczególnych obiektów. Jest to niezmiernie ważne, gdyż znaczna część tworzonych aplikacji przez silnik Unity3D zawiera obiekty animowane, których wydajność wpływa na ogólne wrażenia z uruchamianej aplikacji. Na bardzo podobnym poziomie z animacjami radzą sobie przeglądarki Opera 45 oraz Google Chrome 58. Najgorzej na tym polu wypada przeglądarka wydana przez firmę Microsoft, czyli Edge 38. Inna sytuacja jest z elementami graficznymi, gdyż jak widać w tabeli przeglądarka Firefox 52 jest tą, która najgorzej radzi sobie z tego typu elementami, natomiast wszystkie przeglądarki konkurencyjne wyglądają na tym polu lepiej.

Jednym z elementów, który miał wpływać na wydajność technologii Unity WebGL miał być język asm.js, do którego kompilowany jest finalny kod. Przeglądarka Firefox 58, której autorzy są jednocześnie współautorami tego rozwiązania osiągnęła w badaniu najlepszy wynik, co może powodować, że użycie tej technologii może wpłynąć na wydajność technologii Unity WebGL w przeglądarce internetowej. Technologia te jest również częściowo wspierana przez inne przeglądarki, dlatego ich wyniki nie są znacznie gorsze.

Podsumowując otrzymane wyniki można stwierdzić, że dla różnych przeglądarek wyniki były na bardzo podobnym poziomie, lecz póki co, pod względem wydajnościowym technologia Unity3D odstaje nieco od aplikacji przekonwertowanych do kodu natywnego.

Literatura

[1] A. Evans, M. Romeo, A. Bahrehmand, J. Agenjo i J. Blat, 3D graphics on the web: A survey, 2014.

[2] J.-M. Vanhatupa, On the Development of Browser Games - Technologies of an Emerging Genre, w 2011 7th International Conference on Next Generation Web Services Practices, Salamanca, Hiszpania, 2011.

[3] L. Yijun, C. Wenbin i H. Xiaoman, 3D Graphics Engine Technology Research and Implementation, w 2010 3rd IEEE International Conference, Chengdu, Chiny, 2010.

[4] B. Cowan i B. Capralos, A Survey of Frameworks and Game Engines for Serious Game Development, w 2014 IEEE 14th International Conference on Advanced Learning Technologies , Ateny, Grecja, 2014.

[5] A. Taivalsaari i T. Mikkonen, The Web as an Application Platform: The Saga Continues, w 2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications, Oulu, Finlandia, 2011.

[6] R. Nazarov i J. Galletly, Native browser support for 3D rendering and physics using WebGL, HTML5 and Javascript, w 6th Balkan Conference in Informatics, BCI-LOCAL 2013, Saloniki, Grecja, 2013.

[7] M. Geig, Unity: przewodnik projektanta gier, Gliwice: Helion, 2015.

[8] https://docs.unity3d.com/Manual/IL2CPP.html. [17.06.2017]

[9] https://docs.unity3d.com/Manual/webgl.html. [17.06.2017]

[10] https://blogs.unity3d.com/2016/09/20/understanding-memory-in-unity-webgl/. [17.06.2017]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 174-178
Wysłane: 2017-06-24
Przyjęte: 2017-06-29

Analiza porównawcza interfejsów mobilnych usług do wyszukiwania połączeń transportu publicznego

Filip Skowroński*, Marek Miłosz
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20- 618 Lublin, Polska

Streszczenie: Na podstawie wniosków sformułowanych przez Nielsena, norm ISO, a także metryk wydajnościowych, bazujących na problemach i ocenach użytkowników zostały ocenione i porównane mobilne interfejsy trzech aplikacji służących do wyszukiwania połączeń transportu publicznego. Badania zostały przeprowadzone przy pomocy gogli eyetrackingowych oraz ankiet na pięcioosobowej grupie studentów Politechniki Lubelskiej. Testowane były te same funkcjonalności.

Słowa kluczowe: analiza porównawcza; interfejs mobilny użytkownika; eyetracking; testowanie użyteczności

*Autor do korespondencji.

Adres/adresy e-mail: skowronskifilip@gmail.com

Comparative analysis of mobile interfaces services to search public transport connections

Filip Skowroński*, Marek Miłosz
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract: Based on Nielsen's conclusions, ISO standards, and metrics efficiency, problem-based and user ratings, mobile interfaces of three applications for searching public transport connections were evaluated and compared. The research was conducted using an eye tracking goggle and surveys on a group of five students of the Lublin University of Technology. The same functionalities of applications were tested.

Keywords: comparative analysis; mobile user interfaces; eye tracking; usability testing

*Corresponding author.

E-mail address/addresses: skowronskifilip@gmail.com

1. Wprowadzenie

Interfejsy mobilne pojawiły się wraz z rozprzestrzenieniem telefonów komórkowych [1]. W obecnych czasach urządzenia mobilne umożliwiają funkcjonalności bardzo zbliżone do urządzeń desktopowych, z uwzględnieniem ograniczonej mocy obliczeniowej ze względu na ich rozmiar [2]. Dlatego też interfejs użytkownika dla urządzeń mobilnych stał się równie ważny co interfejs desktopowy. Projektanci interfejsów mobilnych muszą dbać o odpowiedni rozmiar elementów interfejsu (mały ekran), a przy tym powinny być one czytelne, wysoce intuicyjne i jednocześnie atrakcyjne wizualnie.

Aby przeprowadzić analizę porównawczą interfejsów mobilnych należy skorzystać z testów użyteczności pozwalających na obiektywną ocenę interfejsu użytkownika. Głównym celem badań było wykorzystanie testów użyteczności i wybranych metryk [3] do pozyskania danych pozwalających na analizę porównawczą trzech aplikacji mobilnych usług do wyszukiwania połączeń transportu publicznego – *Jakdojade*, *MobileMPK* i *Transportoid* i wyłonienia najlepszej z nich. W tym celu został wykorzystany telefon komórkowy, gogle do śledzenia ruchu gałek ocznych oraz przeprowadzone zostały ankiety w których badani mieli ocenić każdy z interfejsów.

2. Materiały i metody

W przeprowadzonym badaniu wykorzystane zostały zagadnienia użyteczności sformułowane przez Nielsena [4] i normy ISO [5], wzbogacone o metryki wydajnościowe, bazujące na problemach i ocenach użytkowników. Aby zachować warunki mobilności, badania przeprowadzone zostały przy pomocy gogli do eyetrackingu – SMI ETG 2.0 na telefonie komórkowym – Samsung Galaxy S2. Badania przeprowadzone zostały na Politechnice Lubelskiej na pięciu studentach piątego roku informatyki tejże uczelni co pozwoliło na spełnienie minimalnej liczby pozwalającej na przeprowadzenie testów użyteczności [6] [7]. Studenci w większości nie znali testowanych aplikacji (tab. 1).

Badania polegały na wykonaniu 4 scenariuszy w trybie online i offline:
Scenariusz 1 – Sprawdź jak dojechać do Szpitala Chodźki z miejsca w którym się obecnie znajdujesz.
Scenariusz 2 – Sprawdź jak dojechać z ul. Nadbystrzycka 35 na ul. Mełgiewska 12.
Scenariusz 3 – Sprawdź ile razy przejeżdża linia nr 14 na przystanku Skrzetuskiego między godziną 14:00 a 15:00 jadący w kierunku Felin.

Scenariusz 4 – Zmień miasto dla którego będziesz szukał/a połączeń na Gdańsk.

Po wykonaniu wszystkich scenariuszy w danej aplikacji użytkownicy musieli w skali od 1 do 5 odpowiedzieć na poniższe pytania:

- bezproblemowość wykonania zadania
- łatwość w użyciu interfejsu
- czytelność interfejsu
- wygląd interfejsu
- ocena ogólna aplikacji

Dodatkowo zostali poproszeni o udzielenie odpowiedzi na następujące pytania:

- zmienił/a byś coś w interfejsie aplikacji?
- czy masz jakieś dodatkowe uwagi, spostrzeżenia co do interfejsu?

Po skończeniu badania użytkownicy mieli udzielić odpowiedzi na ostatnie pytanie – która aplikacja była według Ciebie najlepsza?

Zebrane przy pomocy gogli do eyetrakckingu materiały zostały następnie skonwentrowane dzięki programowi SMI BeGaze 3.7.42, dzięki czemu możliwe było dokładne obliczenie czasu wykonywanych zadań, a także ustalenie rodzaju i liczby popełnionych błędów podczas badania.

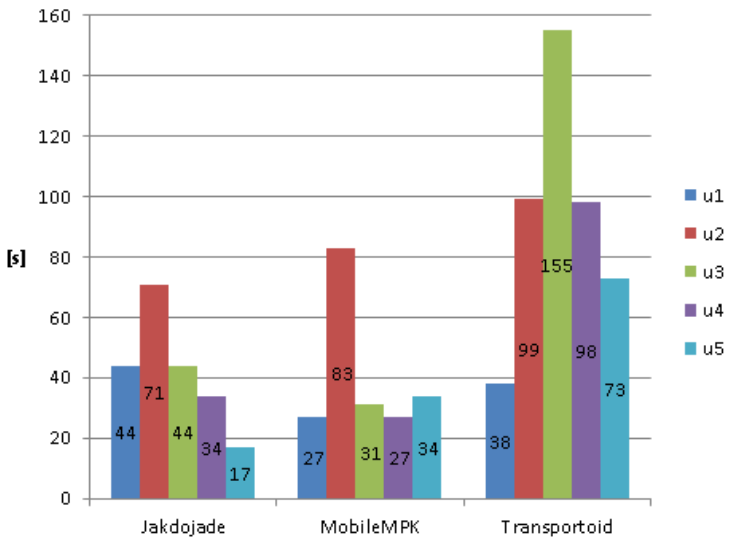
3. Rezultaty

W tabelach 1 – 3 i na rysunkach 1 – 5 zostały zestawione wyniki przeprowadzonych badań. Na rysunkach 6 i 7 został pokazany błąd wyboru złego elementu interfejsu (błąd #5 z tabeli 3). Wyróżnione błędy i liczby ich występowania zostały zawarte w tabeli 3. Znak 'x' w tabeli 3 oznacza, że błąd o konkretnym numerze (od #1 do #5) został popełniony w konkretnej aplikacji z podziałem na użytkownika (u1 – u5). Litery *J, M, T* oznaczają odpowiednio aplikacje

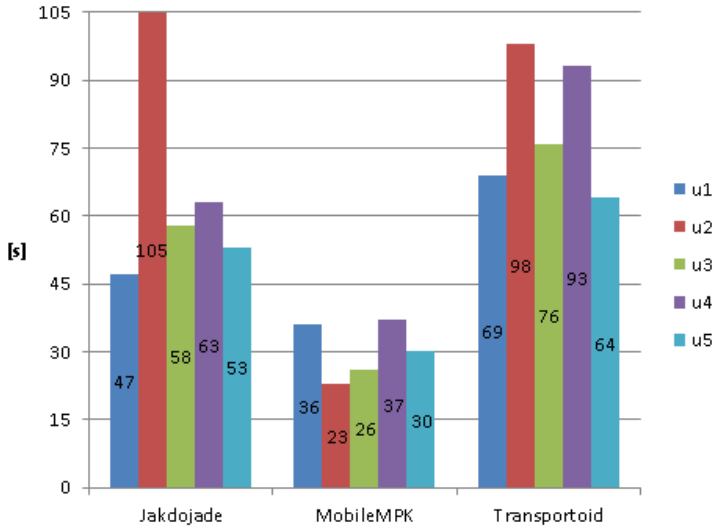
- *jakdojade*
- *mobileMPK*
- *transportoid*

Tabela 1. Wyniki badań przedstawiająca stopień znajomości aplikacji a także liczbę popełnionych błędów

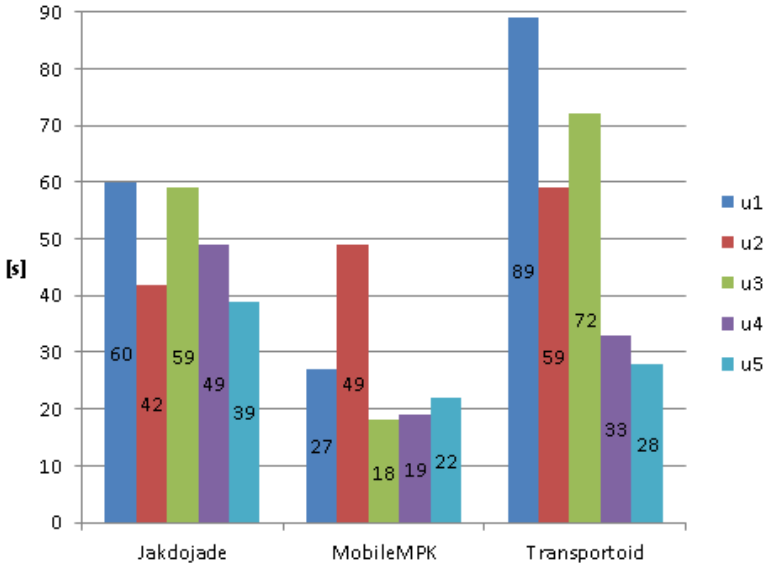
		Osoba 1	**Osoba 2**	**Osoba 3**	**Osoba 4**	**Osoba 5**
Znajomość aplikacji	**Transportoid**	NIE	NIE	NIE	NIE	NIE
	mobileMPK	TAK, PRAWIE CODZIENNIE	NIE	NIE	NIE	TAK, CZĘSTO
	Jakdojade	TAK, DLA INNEGO MIASTA	TAK, RAZ	TAK, RAZ	NIE	NIE
Online						
Liczba błędów		2	9	36	12	10
Offline						
Liczba błędów		–	9	–	–	1



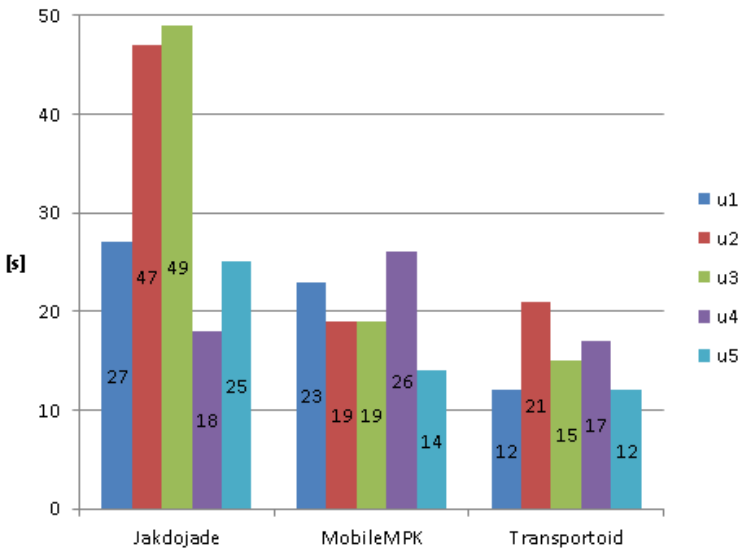
Rys. 1. Czas wykonanania scenariusza 1 – online, przez każdego z badanych (u1 – u5)



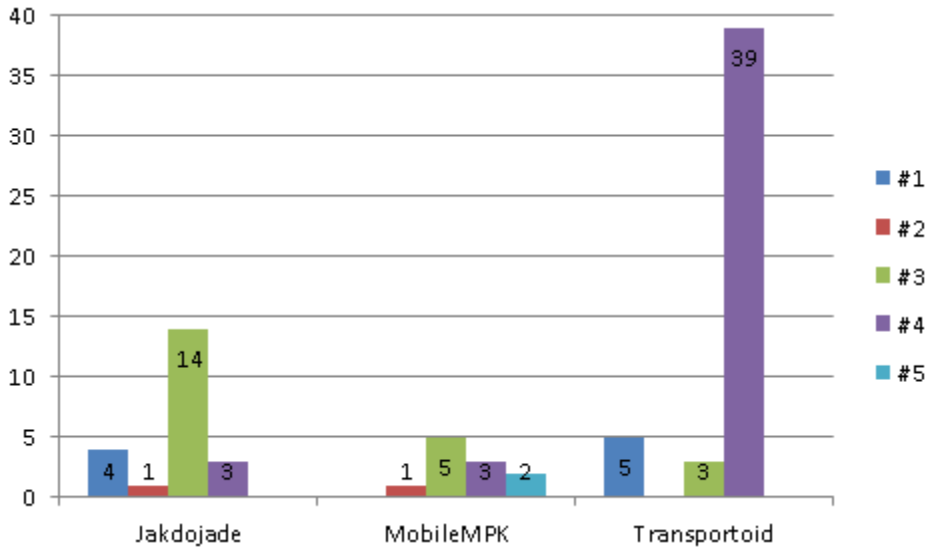
Rys. 2. Czas wykonanania scenariusza 2 – online, przez każdego z badanych (u1 – u5)



Rys. 3. Czas wykonanania scenariusza 3 – online, przez każdego z badanych (u1 – u5)



Rys. 4. Czas wykonanania scenariusza 4 – online, przez każdego z badanych (u1 – u5)



Rys. 5. Łączna liczba popełnionych błędów z podziałem na ich rodzaj (#1 – #5)

Tabela 2. Zsumowane wyniki odpowiedzi użytkowników (u1 – u5) na pytania końcowe w skali od 1 do 5

	Jakdojade	**MobileMPK**	**TransportoID**
Wyniki (punkty)	92	116	82

Tabela 3. Rodzaje błędów

Rodzaj błędu	U1			U2			U3			U4			U5			Suma
	J	M	T	J	M	T	J	M	T	J	M	T	J	M	T	
#1	x		x			x				x			x			5
#2				x	x											2
#3				x		x	x				x		x	x		6
#4					x	x	x	x	x			x		x	x	8
#5								x			x					2

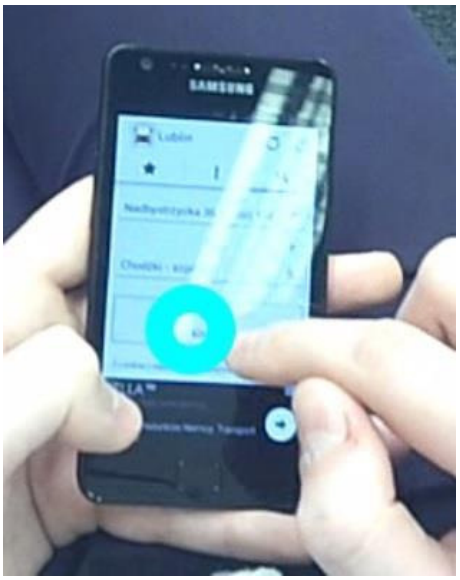
#1 Problem z odnalezieniem odpowiedniego elementu interfejsu służącego do wyszukania rozkładu godzinowego linii na przystanku.

#2 Problem z odnalezieniem odpowiedniego elementu interfejsu służącego do wyboru lokalizacji GPS.

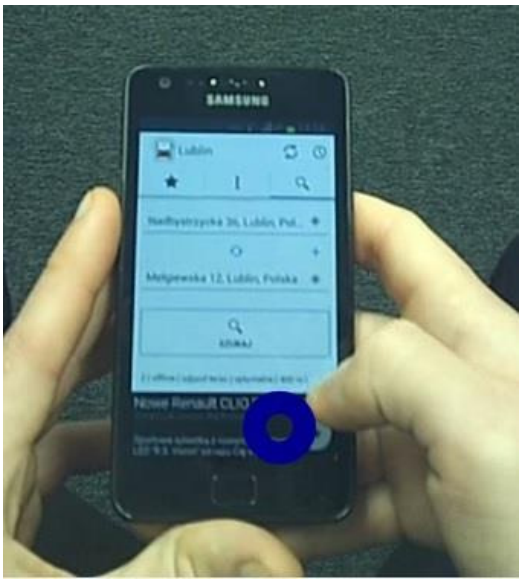
#3 Problem z odnalezieniem odpowiedniego elementu interfejsu służącego do zmiany miasta.

#4 Problem z odnalezieniem odpowiedniego elementu interfejsu służącego do wyszukiwania trasy.

#5 Otwarcie reklamy



Rys. 6. Błąd #5 (1)



Rys. 7. Błąd #5 (2)

4. Wnioski

Uzyskane wyniki badań eyetrackingowych pokryły się z późniejszymi subiektywnymi opiniami uczestników na temat każdego z interfejsów. Pomocne okazały się wybrane metryki zawarte w rozdziale 3, bez których ocena nie byłaby miarodajna. Różnica punktów (tab. 2) pomiędzy interfejsem najlepiej ocenionym, a pozostałymi była na tyle wysoka, że pozwoliła wyłonić najlepszy interfejs.

Badania ze względu na ograniczenia dostępnego sprzętu zostały przeprowadzone na telefonie Samsung Galaxy S2 (2011 rok) co mogło mieć wpływ na otrzymane wyniki czasowe badań.

Pracę nad bardziej kompleksowym porównaniem testowanych interfejsów można by rozszerzyć o przeprowadzenie badań na nowszych telefonach, nie ograniczać się do jednego urządzenia i zebrać większą liczbę osób oceniających interfejsy. Sam sposób przeprowadzonych badań przy pomocy gogli do eyetrackingu nie powinien ulec zmianie bo jest to bardzo wygodne rozwiązanie, pozwalające spojrzeć na poruszanie się po interfejsie oczami użytkownika. Badania zostały przeprowadzone z punktu widzenia nowego użytkownika ze względu na to, że użytkownicy w większości stykali się z interfejsem aplikacji po raz pierwszy. Wyniki czasowe mogłyby zostać skrócone, a liczba wykonanych błędów zredukowana, jeśli użytkownicy zapoznaliby się z interfejsem.

Temat oceny użyteczności interfejsów mobilnych za pomocą mobilnego urządzenia do eyetrackingu okazał się nie być skomplikowany, a otrzymany materiał ułatwiał cały proces porównawczy relacji oko-umysł [8].

Literatura:

[1] G. Santucci : Encyclopedia of Database Systems, pp. 1751-1755. Mobile Interfaces. Springer US, 2009.

[2] Lim MJ., Lee ES., Kwon YM.: Implementing Mobile Interface Based Voice Recognition System. In: Kim T., Ma J., Fang W., Zhang Y., Cuzzocrea A. (eds) Computer Applications for Database, Education, and Ubiquitous Computing. Communications in Computer and Information Science, vol 352. Springer, Berlin, Heidelberg, 2012.

[3] Miłosz M.: Ergonomia systemów informatycznych, Politechnika Lubelska, 2014.

[4] Nielsen, J.: Usability engineering. Morgan Kaufmann Pub., 1994.

[5] ISO 9241 -11:1998: Ergonomics Requirements for Office Work with Visual Display Terminals (VDTs) Part 11: Guidance on usability, 1998.

[6] J. Nielsen, and T. Landauer, A mathematical model of the finding of usability problems. CHI '93 Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, ACM 1993.

[7] Borys M., Plechawska-Wójcik M.: Badanie użyteczności oraz dostępności interfejsu w aplikacjach mobilnych. Nierówności Społeczne a Wzrost Gospodarczy, 2012 vol. 35.

[8] Chynał P., Szymański J.M., Sobecki J.: Using Eyetracking in a Mobile Applications Usability Testing. Lecture Notes in Computer Science, 2012 vol. 7198.

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 179-186
Wysłane: 2017-06-27
Przyjęte: 2017-06-29

Ocena użyteczności współczesnych narzędzi IT w analizie danych Big Data

Rafał Wasiluk*, Dr inż. Piotr Muryjas
[a] Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W niniejszej publikacji poruszono temat współczesnych narzędzi IT w analizie danych Big Data. Przeprowadzone badania dotyczyły liderów rankingu Gartnera z 2017 roku, czyli Tableau Desktop, Qlik Sense i Microsoft Power BI. Ocena użyteczności tych narzędzi, została przeprowadzona zgodnie z wybranymi kryteriami, m. in. ceną i rodzajem licencji, rodzajami połączeń z bazą czy obsługiwanymi platformami operacyjnymi.

Słowa kluczowe: analiza danych Big Data; Tableau; Qlik; Microsoft Power BI

*Autor do korespondencji.

Adres/adresy e-mail: wasiluk.rafal@gmail.com

The assessment of usefulness modern IT tools of data analysis Big Data

Rafał Wasiluk*, Dr inż. Piotr Muryjas
[a] Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This publication discusses the topic of contemporary IT tools in Big Data data analysis. The research involved the leaders from Gartner 2017, Tableau Desktop, Qlik Sense and Microsoft Power BI programs. The utility rating of these tools has been evaluated according to the selected criteria, among others. The price and type of licenses, types of database connections, and supported operating platforms

Keywords: data analysis; Big Data; Tableau Desktop; Qlik Sense; Microsoft Power BI

*Corresponding author.

E-mail address/addresses: wasiluk.rafal@gmail.com

1. Wstęp

W dzisiejszych czasach codziennie masowo generują się duże ilości różnorodnych danych cyfrowych. Ten wzrost danych spowodował konieczność wprowadzenia nowych technik analizy danych, a także wywarł duży wpływ na działalność zarówno małych, średnich, jak i dużych przedsiębiorstw. Zarządy i rady nadzorcze wielu firm sięgają po narzędzia IT, które są w stanie usprawnić działania.

Pojęcie Big Data na przestrzeni lat interpretowane było wielorako. To zagadnienie najczęściej opisywano przez pryzmat cech, z których się składa. Początkowo o Big Data mówiło się tylko w oparciu o koncepcję trzech cech w modelu „3V", takich jak objętość (ang. *volume*), różnorodność (ang. *variety*) i szybkość przetwarzania (ang. *velocity)*. Z kolejnymi latami pojęcie Big Data zaczęło być definiowane w kontekście kolejnych cech: zmienności (ang. *variability)* i złożoności (ang. *veracity)*, wartości informacyjnej (ang. *value)* oraz wizualizacji (ang. *visualization*).[1]

Bankowość, telekomunikacja, turystyka, ubezpieczenia, e-biznes i energetyka – to tylko niektóre z branż, w jakich współcześnie obecna jest analiza Big Data. W przypadku banków, nowoczesne narzędzia pozwalają na przykład na zbadanie wieku klientów, najczęściej korzystających z kart kredytowych. Jeżeli chodzi o telekomunikację, analiza danych Big Data pozwala choćby na zbadanie średnich miesięcznych rachunków klientów, którzy zrezygnowali z usług.

Na rynku można znaleźć bardzo dużo programów służących analizie, jednak celem pracy badawczej, opisanej w niniejszym artykule, była ocena tych narzędzi IT, które znajdują się na pierwszych miejscach w najnowszym rankingu Gartnera z 2017 roku [2]. Oceniono użyteczność Tableau Desktop, Qlik Sense i Microsoft Power BI, które dodatkowo zajmują czołowe miejsca wśród najbardziej wymaganych programów przez pracodawców na portalu Pracuj.pl.

Pierwsza część dotyczy obszarów zastosowań Big Data i wyzwań dzisiejszego świata dla wielkich danych. Pojęciem pokrewnym do analizy danych Big Data jest gromadzenie danych, związane z takimi zagadnieniami jak hurtownie danych, Hadoop i NoSQL. Temu zagadnieniu poświęcono drugi artykuł. Z kolei trzeci artykuł dotyczył współczesnych narzędzi IT do analizy danych. Opisano w nim wybrane programy, a także przedstawiono kryteria i metody oceny. W ostatniej części zaprezentowano ocenę użyteczności wybranych programów.

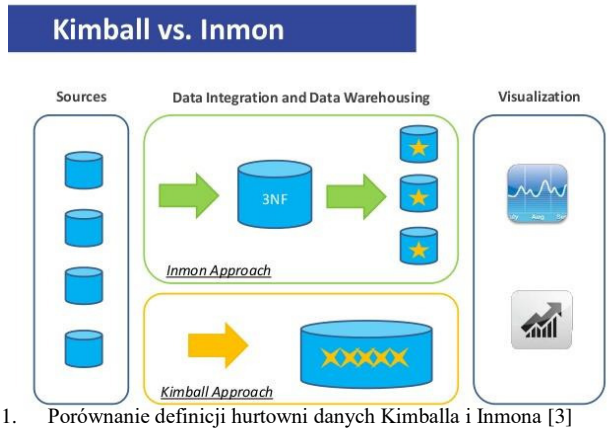
Celem badań była ocena użyteczności programów do analizy danych Big Data: Tableau Desktop, Qlik Sense i Microsoft Power BI zgodnie z przyjętymi kryteriami, m.in. takimi jak cena i rodzaj licencji, rodzaje połączeń z bazą, obsługiwane platformy operacyjne, czy możliwość analizy ad-hoc, in-memory czy w chmurze. Celem oceny było wskazanie najlepszego narzędzia IT do analizy danych z punktu widzenia, młodego analityka, kończącego studia i wchodzącego na rynek pracy.

2. Gromadzenie danych

Zanim nastąpi proces analizy danych Big Data, wcześniej wielkie dane gromadzone są w hurtowniach danych. Z tym zagadnieniem wiążą się także pojęcie Hadoop i model bazy danych NoSQL.

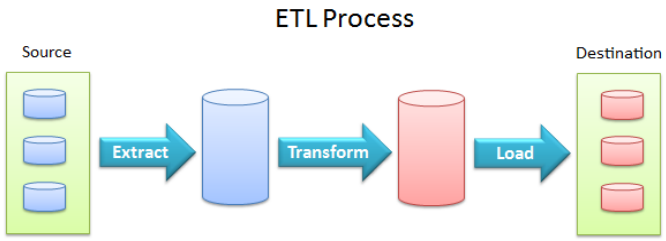
2.1. Hurtownie danych

Hurtownie danych, czyli *data warehouses,* w literaturze opisywane są zarówno w szerszym, jak i węższym ujęciu. Dla Inmona jest to jedynie baza danych o określonych cechach, podczas gdy Kimbal wskazuje obecność dodatkowych elementów, takich jak ekstrakcja danych, ich analiza i narzędzia lub ich zestawy, umożliwiające realizację tych zadań (rys 1.).



Rys. 1. Porównanie definicji hurtowni danych Kimballa i Inmona [3]

Hurtownie danych zasilane są w procesie ETL, którego zadaniem jest integracja danych, pochodzących z różnych źródeł. Ten proces wizualizuje rys. 2.



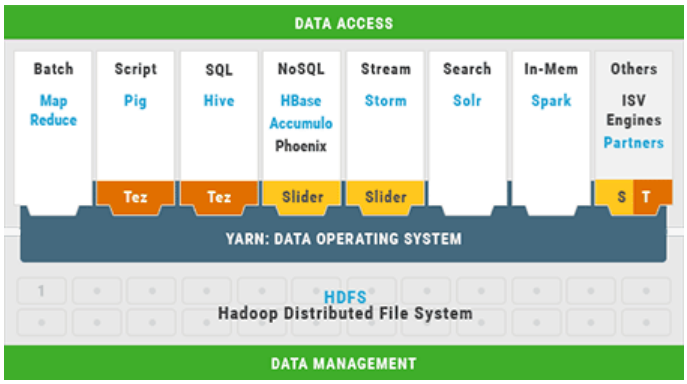
Rys. 2. Proces ETL [4]

Na pojęcie ETL składają się trzy procesy: pobieranie (ang. *extract*), przekształcanie (ang. *transform*) i ładowanie (ang. *load*) [4].

2.2. Hadoop

Przed zdefiniowaniem pojęcia Hadoop, warto wcześniej wspomnieć o MapReduce – systemie, przeznaczonym do tworzenia aplikacji, działających jednocześnie na tysiącach komputerów. Hadoop jest jednym z narzędzi Big Data, służącym do przetwarzania danych, głównie w postaci nieustrukturyzowanej. Apache Hadoop został napisany w języku Java. Hadoop, jako nazwa, stał się synonimem rodziny aplikacji do pracy z Big Data. Apache Hadoop tworzą 4 części: biblioteki bazowe Hadoop, Hadoop File System, Yet Another Resource Negotiator i Hadoop MapReduce. [5]

Wśród najbardziej znanych firm, korzystających z Hadoop, są Cloudera, Hortonworks i MapR. Na rysunku 3 przedstawiono wykorzystanie Apache Hadoop przez firmę Hortonworks [5].



Rys. 3. Wykorzystanie Apache Hadoop przez firmę Hortonworks [5]

Z roku na rok Hadoop znajduje swoje zastosowanie u coraz większej liczby klientów, głównie wśród gigantów branży informatycznej, takimi jak Facebook, Google, IBM, YAHOO!, Amazon, Twitter czy LinkedIn [6].

2.3. Model bazy danych NoSQL

NoSQL, czyli Not Only SQL, jest nierelacyjnym modelem bazy danych. Jego zastosowanie ma ze swojej definicji pozwolić na lepsze skalowanie, niż sam SQL. Mechanizm przechowywania danych NoSQL to „magazyn" danych, zapewniający przechowywanie i pobieranie danych w nieograniczony sposób [7]. Różni się od powszechnie stosowanego modelu relacyjnego. Dzięki jego wdrożeniu można osiągnąć wiele korzyści, takich jak poprawa zrozumienia danych, możliwość gromadzenia danych niestrukturalnych, skalowalność i elastyczność bazy oraz potencjalna możliwość tworzenia unikalnych modeli biznesowych. Porównanie NoSQL i SQL przedstawia tabela 1.

Tabela 1. Porównanie NoSQL i SQL (opracowanie własne)

	NoSQL	SQL
Przechowywanie	Obejmuje wiele typów baz danych	Relacyjna baza danych
Elastyczność	Schematy dynamiczne, informacja aktualizowana na bieżąco	Zapis zgodny z ustalonym schematem, brak możliwości aktualizacji danych dynamicznie
Zgodność ACID	Większość systemów posiada zgodność ACID	Domyślnie zgodne z ACID
Dostęp	Dostępny w dobrze zdefiniowanych wzorach	Brak wzorców, skupia się na definicji indeksu

Za przykład bazy danych, opartej na modelu NoSQL, może posłużyć Cassandra, HBase, Mongo DB, Voldemort, Riak i CouchDB [8].

3. Badania

Przedmiotem badań były współczesne narzędzia IT do analizy danych. Do przeprowadzenia porównania użyto trzech programów, zajmujących czołowe miejsca w najnowszym raporcie Gartnera, przedstawiającym najlepsze narzędzia Business Intelligence i platformy analityczne w lutym 2017 roku [2].

Rys. 4 Raport Gartnera – Magic Quadrant for Business Intelligence and Analytics Platforms [2]

Do oceny wybrano trzy programy: Tableau Desktop, Qlik Sense i Microsoft Power BI.

Tableau Desktop to narzędzie do analizy danych, które pomaga firmom wizualizować i zrozumieć swoje dane. W zależności od potrzeb, użytkownicy Tableau mogą wybrać zarówno darmową, jak i płatną wersję programu. To narzędzie uważane jest za standard w tworzeniu projektów wizualizacyjnych BI. Opis oferty Tableau znajduje się w tabeli 2 [9].

Tabela 2. Oferta Tableau [10]

Typ	Opis
Tableau Desktop	Pozwala na generowanie wykresów w trybie live lub offline. Możliwe jest tworzenie raportów na urządzenia mobilne.
Tableau Server	Zapewnia kompatybilność oraz pełną obsługę chmury m.in. na Microsoft Azure.
Tableau Online	Wersja bezinstalacyjna /przeglądarkowa. Pozwala na publikowanie dashboardów w chmurze między zespołami i dla klientów.
Tableau Mobile	Wersja na smart fony i tablety, głównie do prezentowania gotowych kokpitów menadżerskich.
Tableau Public	Wersja na przeglądarki internetowe, służy do publikowania dashboardów na stronach internetowych.

Qlik Sense jest intuicyjnym systemem, którego prosty interfejs pozwala na tworzenie interaktywnych wizualizacji danych, bez skryptów czy złożonych zapytań SQL, dzięki czemu użytkownicy Qlik Sense nie muszą prosić o pomoc w specjalistów. To narzędzie pozwala na ujednolicenie źródeł danych, tworząc je bardziej czytelnymi. Qlika można używać w dowolnym miejscu z dowolnego urządzenia. Doświadczeni programiści mogą rozszerzać funkcjonalność tego systemu dzięki otwartemu API, a to pozwala im na nieograniczone działanie w nowych obszarach biznesowych (tabela 3) [11].

Tabela 3. Oferta Qlika [12]

Typ	Opis
Qlik Sense	Pozwala na analizę danych, budowanie elastycznych i interaktywnych wizualizacji, ułatwiających podejmowanie ważnych decyzji.
Qlik Sense Cloud	Zapewnia bezpieczny hosting oraz interakcję z Qlik Sens Enterprise, przy pracy nad wspólnym projektem.
Qlik View	Program mniej zaawansowany niż Qlik Sense. Umożliwia tworzenie biznesowych dashboardów, kontrolowanie ich i zarządzenie nimi.
Qlik Analytics Platform	Platforma do umieszczania gotowych dashboardów na dowolnych urządzeniach, zarówno mobilnych, jak i stacjonarnych.
Qlik NPrinting	Pozwala na łatwe i szybkie tworzenie raportów, w różnych popularnych formatach, korzystając z analizy Qlik Sense i Qlik View.
Qlik Data Market	Pomaga znaleźć, podłączyć i zarządzać danymi ze źródeł zewnętrznych.

Microsoft Power BI służy do analizy, wizualizacji oraz udostępniania danych w formie gotowych raportów. Ponadto, pozwala na tworzenie kokpitów menadżerskich, a także umożliwia bezpośrednie podłączenie do danych. Zbiór funkcji i usług online w Microsoft Power BI jest zorientowany w chmurze, przez co użytkownicy mają do nich dostęp w dowolnym miejscu i w dowolnym czasie (tabela 4) [13].

Tabela 4. Oferta Microsoft [14]

Typ	Opis
Power BI Desktop	Darmowy program, służący do tworzenia zaawansowanych raportów oraz publikowania ich w usłudze BI.
Power BI Pro	Płatny program, mający więcej możliwości niż darmowa wersja Power BI Desktop (m.in. w wersji rozszerzonej limity danych są zwiększone do 10 GB)

Tableau Desktop, Qlik Senes i Microsoft Power BI są narzędziami IT, przeznaczonymi nie tylko dla domowych użytkowników, ale także małych, średnich, a nawet dużych przedsiębiorstw. Każdy z tych użytkowników ceni sobie konkretne funkcje w wymienionych programach, które z jego punktu widzenia będą użyteczne. Użyteczność, o której mowa, dla każdego użytkownika wybranych programów, wiąże się z innym pojęciem.

3.1. Metody i przebieg badań

Do oceny opisanych narzędzi IT (Tableau Desktop, Qlik Sense i Microsoft Power BI) metodą porównawczą, wybrano osiem kryteriów. Lista wykorzystanych kryteriów oraz opis użytych metod została przedstawiona w tabeli 5.

Tabela 5. Kryteria oceny użyteczności wybranych programów

Lp.	Kryteria oceny	Metody
1.	Cena i rodzaj licencji	Sprawdzenie i zestawienie cen programów w celu zbadania, który z nich jest najdroższy, a który najtańszy.
2.	Rodzaje połączeń z bazą	Sprawdzenie i zestawienie wybranych programów pod względem ilości i rodzajów obsługiwanych baz danych.
3.	Obsługiwane platformy operacyjne	Sprawdzenie na jakich platformach operacyjnych obsługiwane są wybrane programy i zestawienie wyników.
4.	Target firmy	Sprawdzenie dostępności produktów zarówno dla dużych, jak i małych i średnich firm.
5.	Narzędzia do przygotowywania danych	Sprawdzenie i zestawienie informacji, dotyczących wbudowanych narzędzi do przygotowywania danych na wybranych programach.
6.	Dokumentacja	Sprawdzenie dostępu do samouczków i dokumentacji producenta. Zbadanie dostępu do forów użytkowników.
7.	Funkcje dla wysoko wykwalifikowanych użytkowników	Sprawdzenie i zestawienie wybranych narzędzi IT pod względem obecności zaawansowanych funkcji (analizy lub rozszerzalności narzędzi).
8.	Analiza ad-hoc, in-memory oraz w chmurze	Sprawdzenie, czy wybrane narzędzie ma możliwość analizy ad-hoc i analizy in-memory. Zestawienie wybranych programów pod względem możliwości analiz Business Intelligence w chmurze.

Użyteczność danego programu, dla konkretnego użytkownika, oznacza co innego. Przykładowo, właściciel jednoosobowej firmy, może skorzystać z funkcji tworzenia interaktywnych raportów, które następnie może przedstawić na dowolnym urządzeniu mobilnym podczas spotkania z klientem. Taki użytkownik nie ma potrzeby zarządzania dużymi wolumenami danych czy wynajmowaniu serwerowni, w odróżnieniu do dużych przedsiębiorstw. Zarządy czy rady nadzorcze korporacji cenią sobie programy, które są w stanie wykonać analizy danych Big Data w szybkim tempie. Ponadto, zwracają także uwagę na brak jakichkolwiek ograniczeń, przykładowo w ilości danych. Dlatego też, przy ocenie użyteczności wybranych narzędzi IT, na pierwszym planie stawiano użytkownika i jego potrzeby, które determinują pojęcie użyteczności Tableau Desktop, Qlik Sense i Microsoft Power BI.

3.2. Wyniki badań

Cena i rodzaj licencji

Pierwszym kryterium, zgodnie z którym dokonano oceny użyteczności wybranych programów, była cena i rodzaj licencji. Badania skupiły się na określeniu różnic między ceną a czasem licencji w poszczególnych wersjach programu. Dane, dotyczące ceny i rodzaju licencji Tableau, obrazuje tabela 6.

Tabela 6. Tableau – licencja abonamentowa [10]

Wersja	Cena	Liczba użytkowników	Czas licencji
Tableau Desktop	-	1	14 dni
Tableau Desktop	35$	1	1 miesiąc
Tableau Desktop Professional	70$	1	1 miesiąc
Tableau Desktop Professional (akademicka)	-	1	1 rok
Tableau Server	35$	1	1 miesiąc
Tableau Online	42$	1	1 miesiąc

Każda z cen, przedstawionych w tabeli, może być wyznaczona indywidualnie, w zależności od użytkowników, długości trwania umowy czy wielkości baz danych.

Drugim badanym programem, pod względem ceny i rodzaju licencji, był Qlik. Zgodnie z cennikiem licencji, przedstawionym na stronie producenta Qlika, comiesięczny abonament, z którego może skorzystać jeden użytkownik, zależny jest od wersji programu. Przykładowe dane, dotyczące kosztów, związanych z konkretną wersją Qlika, zostały przedstawione w tabeli 7.

Tabela 7. Qlik – licencja abonamentowa [12]

Wersja	Cena	Liczba użytkowników	Czas licencji
Qlik Sense Desktop	-	1	Bez limitu
Qlik Sense Enterprise	Koszt licencji ustalany indywidualnie		
Qlik Sense Cloud Basic	-	1-5	Bez limitu
Qlik Sense Cloud Business	25$	1	1 miesiąc
Qlik View	-	1	Bez limitu
Qlik Analytics Platform	Koszt licencji ustalany indywidualnie		

Produktami darmowymi Qlika są Qlik Sense Desktop, Qlik Sense Cloud Basic i Qlic View. Użytkownicy mogą korzystać z nich bez limitu. Natomiast jeżeli chodzi o Qlik Sense Enterprise i Qlik Analytics Platform, cena jest ustalana indywidualnie, w zależności od czasu licencji i liczby użytkowników. Z kolei z wersji Qlik Sense Cloud Basic, bez limitu czasowego może bezpłatnie korzystać do 5 osób.

Trzecim badanym programem, pod względem ceny i rodzaju licencji, był Microsoft Power BI. Przykładowe dane, dotyczące kwot, związanych z konkretną wersją programu, zostały przedstawione w tabeli 8.

Tabela 8. Microsoft Power BI – licencja abonamentowa [14]

Wersja	Cena	Liczba użytkowników	Czas licencji
Power BI Desktop	-	1	Bez limitu
Power BI Pro	9,99$ / 99,99 $	1	1 miesiąc / 1 rok
Power BI Premium	Koszt licencji ustalany indywidualnie		

Jak wynika z tabeli, licencję na Power BI Desktop użytkownik może uzyskać bezpłatnie na nieograniczony okres. Za Power BI Pro jeden użytkownik musiałby za miesiąc zapłacić 9,99 dolarów, co w skali roku daje 99,99 dolarów. Natomiast koszt licencji do Power BI Premium ustalany jest indywidualnie, w zależności od ceny, liczby użytkowników i czasu licencji.

Rodzaje połączeń z bazą

Wybrane narzędzia IT zbadano także pod względem rodzajów połączeń z bazą. Część informacji na ten temat, odnośnie Tableau Desktop, wizualizuje tabela 9.

Tabela 9. Tableau – wybrane rodzaje połączeń z bazą [10]

Wersja	Rodzaj połączenia
Podstawowa	• Microsoft Excel • Pliki tekstowe (np. pliki .csv) • Microsoft Access for Windows • JSON • Tableau Data Extract
Profesjonalna	• Actian • Amazon • Anaplan • Microsoft • MySQL • Oracle • Hortonworks Hadoop Hive • MapR Hadoop Hive • Pliki tekstowe (np. pliki .csv) • Tableau Data Extract

Jak prezentuje tabela 9, w wersji podstawowej Tableau Desktop użytkownicy mają możliwość skorzystania zaledwie z pięciu rodzajów połączeń, podczas gdy wersja profesjonalna udostępnia znacznie więcej możliwości rodzajów połączeń z bazą.

Qlik, w odróżnieniu do Tableau, niezależnie od wersji programu, korzysta z tych samych rodzajów połączeń z bazą (tabela 10).

Tabela 10. Qlik – wybrane rodzaje połączeń z bazą [12]

Wersja	Rodzaj połączenia
Qlik	Bazy danych: • Apache Hive, • ClouderaImpala, • MySQL Enterprise, • Teradata, • Oracle.
	Pliki: • CSV, • HTML, • XML, • XLSX, • Natywne pliki programu Qlik – QVD i QVX.

Dla każdej wersji Qlika, dostępne są te same rodzaje połączeń, zarówno z bazami, jak i z plikami. Podobnie jest z rodzajami połączeń w przypadku Microsoft Power BI (tabela 11).

Tabela 11. Microsoft Power BI – wybrane rodzaje połączeń z bazą [14]

Wersja	Rodzaj połączenia
Microsoft Power BI	Pliki: • Excel, • Text/CSV, • XML, • JSON, • SharePointFolder.
	Bazy danych: • SQL Server Database, • Access Database, • Azure, • IBM DB2 Database, • Impala.
	Serwisy internetowe: • SharePointOnlineList, • Microsoft Exchange Online, • Dynamics 365 (online), • Salesforce Objects, • Google Analytics.
	Pozostałe: • Web, • OData Feed, • Spark (Beta), • R Script, • ODBC.

Dla każdej wersji Microsoft Power BI są dostępne te same rodzaje połączeń.

Obsługiwane platformy operacyjne

Wszystkie wersje Tableau są dostępne na platformy operacyjne Windows, Mac, Android, iPhone/iPad czy przeglądarki internetowe. W zależności od konkretnej wersji programu (Tableau Desktop, Tableau Server czy Tableau Online), określane są inne wymagane parametry. Dla przykładu, Tableau Desktop i Tableau Online mogą być obsługiwane na platformach operacyjnych iMac/MacBook, a w wersji Tableau Server nie ma takiej możliwości.

W wersji Tableau Desktop minimalne wymagania sprzętowe wynoszą 2 GB RAM, podczas gdy w Tableau Server potrzeba minimum 8 GB RAM. Tableau Desktop nie obsługuje przeglądarek internetowych, podczas gdy zarówno Tableau Server, jak i Tableau Online działa m.in. pod Chrome, Apple Safari, Internet Explorer 11 czy Mozilla Firefox [10].

Qlik z kolei, obsługiwany jest m.in. na systemie operacyjnym Windows 7 lub nowszym. W Qlik Sense wymagany jest co najmniej czterordzeniowy procesor, podczas gdy do Qlik View jedynie dwurdzeniowy. Zarówno Qlik Sense, jak i Qlik View, działają pod Chrome, Apple Safari, Internet Explorer 11 czy Mozilla Firefox. [12]

Natomiast Microsoft Power BI, obsługiwany jest jedynie pod platformami Windows. Wszystkie rodzaje tego narzędzia IT (Microsoft Power BI Pro i Microsoft Power Professional) mają te same wymagania sprzętowe (minimum 4 GB RAM oraz 1,5 GB wolnej pamięci przestrzeni na dysku). Oba programy są obsługiwane przez Internet Explorer 9 lub nowszy [14].

Grupa docelowa firmy

Tableau, kieruje swoje produkty do zarządów i rad nadzorczych w dużych i średnich przedsiębiorstwach z różnych branż (np. bankowość, komunikacja, ubezpieczenia). Tableau tworzy także analizy w innych obszarach, choćby takich jak rząd, edukacja czy organizacje non-profit.

W przypadku Qlika, grupy docelowej firmy są nie tylko duże i średnie przedsiębiorstwa, jak wspomniane było w przypadku Tableau, ale także małe, nawet jednoosobowe firmy.

Microsoft Power BI, podobnie jak Tableau, tworzy swoje produkty z myślą o dużych i średnich przedsiębiorstwach, działających na różnych obszarach i w różnych branżach.

Narzędzia do przygotowywania danych

Zarówno Tableau, jak i Microsoft Power BI nie posiadają wbudowanych narzędzi ETL do przetwarzania danych. Ich transformacje ograniczają się jedynie do zadawania zapytań, w języku zrozumiałym dla źródła. Qlik natomiast posiada wbudowane narzędzia ETL do przetwarzania danych [15, 16].

Dokumentacja

Na stronie producenta Tableau, Qlik oraz Microsoft Power BI, znajdują się odnośniki do dokumentacji, w której zawarte są samouczki oraz przydatne wskazówki. Jedynie Qlik posiada polskojęzyczną wersję dokumentacji. Natomiast wszystkie trzy narzędzia posiadają fora użytkowników w języku polskim.

Funkcje dla wysoko wykwalifikowanych użytkowników

Produkty Tableau dostosowano do wymagań wysoko wykwalifikowanych użytkowników, znających język R. Tableu posiada funkcje takie jak analiza trendów, modelowanie regresji czy korelacje.

Qlik Analytics Platform (QAP), który ma otwarte API aplikacji, pozwala programistom na tworzenie własnych obiektów (pisanych w języku skryptowym JavaScript).

Microsoft Power BI z łatwością tworzy proste wizualizacje. Jednak gdy chodzi o zaawansowane analizy, potrzebny jest ekspert w języku DAX do analizy danych. [17]

Analiza technologii ad-hoc, in-memory i w chmurze

Tableau Desktop pozwala na wykonywanie obliczeń ad-hoc, dzięki połączeniu Live lub Offline. Dzięki połączeniu Live, można np. tworzyć zapytania ad-hoc, mając wciąż aktualizowane zbiory danych. Podczas połączenia Offline wykonywane są obliczenia in-memory.

Tableau posiada analizę danych in-memory. Jest to wykorzystanie pamięci operacyjnej komputera do analiz, dzięki czemu ograniczenia przepustowości sieci przestają wpływać na prędkość analiz (wszystkie analizowane dane zostają pobrane do pamięci podręcznej).

Analiza danych w chmurze, w Tableau, dostępna jest jedynie w wersji Professional i Server. W tańszej wersji programu nie ma możliwości połączenia z chmurą.

Nowa odsłona Qlik Sense oferuje użytkownikom możliwości analizy ad-hoc, in-memory oraz w chmurze. Qlik posiada wbudowany silnik asocjacyjny (QIX), dzięki któremu możliwe jest generowanie nowoczesnych raportów oraz analiza ad-hoc. Każde kliknięcie na wykres spowoduje wykonanie analitycznego obliczenia, takiego jak zwijanie, filtrowanie itp. Qlik obsługuje także analizę in-memory, która dzięki wykorzystaniu pamięci operacyjnej RAM komputera, wykonuje obliczenia. Inwestycja w dużą ilość RAM, pozwoli użytkownikowi na obliczanie milionowych danych w czasie rzeczywistym.

Analizy danych, przy wykorzystaniu pamięci operacyjnej RAM komputera, są dostępne w Power BI Desktop. W każdej wersji programu można wykonywać zarówno analizy ad-hoc, analizy in-memory i analizy w chmurze. [18]

4. Ocena użyteczności wybranych narzędzi IT

Wybrane programy zostały ocenione zgodnie z przyjętymi kryteriami i metodami, w skali od 1 do 5. Celem oceny było wskazanie najlepszego narzędzia IT do analizy danych z punktu widzenia młodego analityka, kończącego studia i wchodzącego na rynek pracy. Tabela 12 prezentuje ocenę użyteczności wybranych programów.

Tabela 12. Ocena użyteczności wybranych programów

	Tableau Desktop (pkt)	Qlik Sense (pkt)	Microsoft Power BI (pkt)
Cena i rodzaj licencji	4	5	5
Rodzaje połączeń z bazą	4	4	5
Obsługiwane platformy operacyjne	4	3	4
Target firmy	4	5	4
Narzędzia do przygotowywania danych	4	5	4
Dokumentacja	5	5	5
Funkcje dla wysoko wykwalifikowanych użytkowników	5	5	4
Analiza ad-hoc, in-memory oraz w chmurze	5	5	5
Podsumowanie:	35	37	36

Cena i rodzaj licencji

W skali od 1 do 5, zarówno Qlik, jak i Microsoft Power BI, oceniono na maksymalną ilość punktów. Tę decyzję uzasadniono tym, że nie trzeba być studentem, żeby mieć darmową wersję programu. Co prawda Tableau udostępnia studentom bezpłatną wersję akademicką Tableau Desktop Professional na rok i daje możliwość przedłużenia licencji po okazaniu legitymacji studenckiej, jednak nie ma on tych samych możliwości, co darmowe wersje Qlika i Microsoft Power BI. Kluczową funkcją, jaka zaważyła w ocenie, był brak możliwości dzielenia się dashboardami w Tableau Desktop Professional.

Rodzaje połączeń z bazą

Pod względem rodzajów połączeń z bazą, Qlik i Tableau uzyskały (4) punkty. Najwyższą notę (5) otrzymał produkt Microsoftu. Ta decyzja została podyktowana tym, że Qlik posiada najmniejszą ilość rodzajów połączeń z bazami danych, w porównaniu z innymi ocenianymi narzędziami. Tableau stracił punkt, gdyż w wersji darmowej ma zaledwie 10 rodzajów połączeń, bez możliwości łączenia się z danymi w chmurze. Zarówno Power BI, jak i Tableau, mają możliwość analizy danych w portalach społecznościowych. Qlik z kolei nie posiada takiej możliwości, za co odjęto mu punkt w powyższej ocenie. Microsoft Power BI w tym zestawieniu prezentuje się najlepiej, gdyż niezależnie od wersji programu, udostępnia te same rodzaje połączeń - z plikami, bazami danych, serwisami internetowymi czy innymi.

Obsługiwane platformy operacyjne

Jeżeli chodzi o obsługiwane platformy operacyjne, żaden z programów nie otrzymał maksymalnej ilości punktów. Tableau i Power BI otrzymały ex aequo (4) punkty zaś Qlik oceniono na (3). Powodem tej decyzji były następujące zależności:

- żaden z wyżej wymienionych programów nie jest dostępny na każdą platformę,
- Tableau i Qlik nie mają wersji na Windows Phone,
- Power BI i Qlik nie mogą być obsługiwane w systemie operacyjnym Mac (są dostępne tylko na iPhone'a/iPada), z kolei Tableau, jako jedyny z nich, posiada wersję na urządzenia dekstopowe Maca.
- Wymagania sprzętowe wszystkich narzędzi były porównywalne (wymagały m.in. 2-4 rdzeniowego procesora, 2-8 GB RAM, ~1.5 GB wolnej przestrzeni na dysku).
- Jeżeli chodzi o obsługiwane przeglądarki, Tableau Desktop w zestawieniu wypadł najgorzej, ponieważ nie obsługuje żadnej. Qlik i Power BI obsługują te same przeglądarki.

Podsumowując, pomimo że Tableau i Power BI otrzymały tyle samo punktów, wybierając Tableau nie trzeba sie zastanawiać czy wszystkie komputery w firmie są pod systemem Windows czy może któryś ma system Mac.

Target firmy

Zestawiając wspomniane narzędzia IT pod względem targetu firmy, Qlik otrzymał maksymalną ilość punktów (5), natomiast Tableau i Power BI ex aequo po 4 punkty. Tableau, podobnie jak Microsoft Power BI, kieruje swoje produkty do dużych i średnich przedsiębiorstw, które posiadają już własne hurtownie danych i narzędzia do ich przetwarzania. Natomiast Qlik tworzy programy przeznaczone głównie do małych i średnich firm, dzięki zastosowaniu wbudowanych narzędzi ETL. Warto zauważyć, że istnieje wersja darmowa Qlik Sense Cloud, która zachęca właścicieli małych przedsiębiorstw do korzystania z tego narzędzia, gdyż oferuje możliwość tworzenia i udostępniania użytkownikom prostych raportów za darmo.

Narzędzia do przygotowywania danych

Wśród ocenianych firm, pod względem narzędzi do przygotowywania danych, najlepiej prezentuje się Qlik, który głównie za to, iż posiada własne narzędzie ETL, dostał 5 punktów. Dwa pozostałe programy w zestawieniu uzyskały o jeden punkt mniej, gdyż są uzależnione od zewnętrznych narzędzi ETL.

Dokumentacja

Wszystkie trzy programy posiadają bardzo bogatą dokumentację, zarówno w języku angielskim jak i polskim. Tableau, Power BI i Qlik mają samouczki w postaci filmików, przykładowe zastosowania programów oraz dodatki, wzbogacające możliwości analitycznie. Tak bogata

oferta sprawiła, że każdy z programów oceniono maksymalną ilością punktów (5).

Funkcje dla wysoko wykwalifikowanych użytkowników

Pod względem obecności funkcji dla wysoko wykwalifikowanych użytkowników, ze wszystkich porównywanych programów, najlepiej wypadły Tableau i Qlik. Oba narzędzia IT uzyskały ex aequo maksymalną ilość punktów (5), gdyż funkcjonalność tych programów można rozszerzać znając język skryptowy (np. java script). Znajomość języka R również pomoże tworzeniu rozbudowanych analiz w przypadku wspomnianych narzędzi. W tym zestawieniu na ostatniej pozycji uplasował się Power BI, który stracił punkt za to, że możliwość korzystania z zaawansowanych funkcji, uzależnił od znajomości specjalnego języka dedykowanego DAX do analizy danych.

Analiza ad-hoc, in-memory oraz w chmurze

Każdy z opisywanych programów, niezależnie od wersji posiada możliwość analizy ad-hoc, in-memory oraz w chmurze. Dlatego zarówno Tableau, Qlik oraz Microsoft Power BI otrzymały w zestawieniu tą samą liczbę punktów (5).

5. Wnioski

Analizując otrzymane wyniki, program Qlik, na tle badanych narzędzi IT, nie pod każdym względem wypadł najlepiej, jednak mimo to, otrzymał najlepszy wynik (37 z 40 możliwych punktów). Microsoft Power BI łącznie dostał 36 punktów. Tym samym zajął drugie miejsce w zestawieniu oceny użyteczności współczesnych narzędzi IT do analizy danych Big Data. Na ostatnim miejscu, różnicą 1 punktu, uplasował się Tableau z wynikiem 35 punktów. Zgodnie z powyższym, najbardziej użytecznym programem do analizy danych, z punktu widzenia studenta i analityka biznesowego, wchodzącego na rynek pracy, okazało się narzędzie Qlik. Warto jednak zaznaczyć, iż różnica między pierwszym a ostatnim miejscem zestawienia, była minimalna. Wszystkie badane programy spełniają kryteria użyteczności pod każdym względem. Te tezę może potwierdzić fakt, iż w żadnym z kryteriów oceniania, nie pojawiła się ocena niższa niż 3 punkty. Hipoteza, postawiona na początku niniejszej pracy, została zweryfikowana pozytywnie.

Literatura

[1] Marz N., Warren J.: Big Data. Najlepsze praktyki budowy skalowalnych systemów obsługi danych w czasie rzeczywistym, Helion, Gliwice, 2016

[2] Howson C., Idoine C., Oestreich T., Richardson J., Sallam R., Tapadinhas J., Magic Quadrant for Business Intelligence and Analytics Platforms, Gartner, https://www.gartner.com/doc/reprints?id=1-3TYE0CD&ct=170221&st=sb., [02.06.2017]

[3] Rangarajan S., Data Warehouse Design – Inmon versus Kimball, http://tdan.com/data-warehouse-design-inmon-versus-kimball/20300 [03.06.2017]

[4] Mudrakola S., Manage your data with these 3 open source etl tools, http://techgenix.com/open-source-etl-tools/ [04.06.2017]

[5] Strona firmy Hortonworks, wykorzystującej Apache Hadoop https://hortonworks.com/products/data-center/hdp/ [21.05.2017]

[6] Siudziński M., Hadoop, czyli przetwarzanie rozproszone w open source, http://itwiz.pl/hadoop-czyli-przetwarzanie-rozproszone-open-source/ [20.05.2017]

[7] Czajkowski L., A Comparison of SQL and NoSQL to Simplify Your Database Decision, https://blog.couchbase.com/comparison-sql-nosql-simplify-database-decision/ [11.06.2017]

[8] Harazin J., Narzędzia Business Intelligence dla managera: QlikView i Qlik Sense, http://www.jcommerce.pl/jpro/article/narzedzia-business-intelligence-dla-managera-qlikview-i-qlik-sense [5.06.2017]

[9] Baker P., Tableau Desktop, http://www.pcmag.com/article2/0,2817,2491943,00.asp, [6.06.2017]

[10] Strona producenta Tableau, https://www.tableau.com, [04.06.2017]

[11] Prichard J., What's the difference between Qlik Sense and Qlik View?, http://optimalbi.com/blog/2016/01/18/whats-the-difference-between-qlik-sense-and-qlik-view/ [3.06.2017]

[12] Strona producenta Qlik, http://www.qlik.com/pl-pl/ [04.06.2017]

[13] Ris O., Microsoft Power BI, PCMag, http://uk.pcmag.com/microsoft-power-bi/74174/review/microsoft-power-bi [26.05.2017]

[14] Strona producenta Microsoft Power BI, https://powerbi.microsoft.com/en-us/ [04.06.2017]

[15] Rui R., Use Power BI Desktop as an ETL Tool, https://ruiromanoblog.wordpress.com/2016/04/21/use-power-bi-desktop-as-an-etl-tool/ [30.05.2017]

[16] Chadran S., Is Microsoft Power BI a data warehouse also?, Olympic Software, https://www.olympic.co.nz/is-microsoft-power-bi-a-data-warehouse-also/ [28.05.2017]

[17] Iseminger D., Measuers in Power BI Desktop, https://powerbi.microsoft.com/en-us/documentation/powerbi-desktop-measures/ [01.06.2017]

[18] Deng S., Saxton A.,.Schiavon A., Wprowadzenie do usługi Microsoft Power BI Embedded, https://docs.microsoft.com/pl-pl/azure/power-bi-embedded/power-bi-embedded-get-started [29.05.2017]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI X (2017) 187-192
Wysłane: 2017-06-28
Przyjęte: 2017-06-29

Analiza zastosowania okularów 3D w symulacji przestrzennej na przykładzie wizualizacji architektonicznej

Mateusz Karol Smaga*, Małgorzata Plechawska-Wójcik
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule opisane zostało przygotowanie wizualizacji architektonicznej wirtualnego muzeum. Miało to na celu zbadanie możliwości płynących z wykorzystania okularów wirtualnej rzeczywistości, a także skanów 3D oraz aplikacji do projektowania. W czasie badań okulary VR pokazały duże możliwości w zakresie przedstawiania architektury. Wyniki testów wydajności potwierdziły duży potencjał płynący z wykorzystania w wizualizacjach skanów 3D i importu modeli z zewnętrznych aplikacji, mimo wymogu ich obróbki i uproszczenia siatki. Testy wykazały jednak niewystarczającą wydajność procesora graficznego w badanym smartfonie, co pokazuje potrzebę rozwoju rynku w tym kierunku. Stworzyłoby to możliwości tworzenia lepszych aplikacji VR.

Słowa kluczowe: rzeczywistość wirtualna; unity; grafika 3D; skany 3D

*Autor do korespondencji.

Adres/adresy e-mail: mateusz.smaga@gmail.com

Analysis of the 3D glasses application in a spatial simulation on the basis of the architectural visualization

Mateusz Karol Smaga*, Małgorzata Plechawska-Wójcik
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The arcticle describes preparation of architectural visualization of the virtual museum. It purpose was to explore possibilities of using VR glasses, 3D scans and design aplications. During the study VR glasses showed great opportunities in the field of architectural visualisation. The results of the performance tests confirmed potencial in 3D scans and models from design applications, although they require mesh prcoessing and simplification. However, the tests showed insufficient performance of tested smartphone's GPU, which shows that market needs to advance in this direction. This would make it possible to create better VR applications.

Keywords: virtual reality; unity; 3D graphics; 3D scans

*Corresponding author.

E-mail address/addresses: mateusz.smaga@gmail.com

1. Wprowadzenie

Temat wirtualnej rzeczywistości (ang. *Virtual reality*, VR) nie jest nowym zagadnieniem. Poczynając od stereoskopów poprzez system CAVE (ang. *Cave Automatic Virtual Environment*) [1], czyli pokoju, na którego ściany wyświetlany jest obraz z rzutników, co tworzy efekt wirtualnej przestrzeni dla osób przebywających wewnątrz pokoju, aż do coraz powszechniej znanych dzisiaj HMD (ang. *Head Mounted Display*). HMD są okularami wirtualnej rzeczywistości składającymi się z ekranu lub ekranów, na których wyświetlane są 2 obrazy odwzorowujące obraz dla poszczególnych oczu. Tworzone jest w ten sposób złudzenie przebywania w trójwymiarowej przestrzeni.

Wirtualna rzeczywistość to zagadnienie wolno rozwijające się ze względu na ograniczenia technologicznie. Sytuacja zmieniła się jednak w ostatnich latach. Na rynku pojawiły się relatywnie tanie (w porównaniu z dostępnymi profesjonalnymi rozwiązaniami) okulary VR - Oculus Rift [2], HTC Vive, jak również Playstation VR dla użytkowników konsol Sony Playstation 4. Ich odpowiednikiem na rynku mobilnym jest między innymi Samsung Gear VR i Google Cardboard. Rozwiązania te zamiast wbudowanego ekranu wykorzystują telefon użytkownika.

Mimo częstego powiązania wirtualnej rzeczywistości z grami wideo wykorzystanie okularów VR znacząco wykracza poza aplikacje typowo rozrywkowe. Jednym z takich rozwiązań jest wizualizacja architektoniczna. Zarówno architektury nam znanej, odtworzonej na podstawie realnego świata, jak i światów stworzonych zupełnie od podstaw. Jednym z wczesnych przykładów jest gra PlastiCity [3]. Przedstawiała ona projekt przebudowy centrum Bradford w Wielkiej Brytanii. Każdy z mieszkańców mógł zwiedzić projekt i na własne oczy zobaczyć przewidywane zmiany.

Inne przykłady to między innymi odtwarzanie stanowisk archeologicznych grobowców [4], projekt odtworzenia rzymskiego teatru w Byblos [5], rekonstrukcja pałacu Ypres Salient w Belgii na podstawie danych historycznych [6], czy sale operacyjne w Walter Reed National Military Medical Center utworzone przy pomocy wielokrotnych wirtualnych spacerów po projektowanej sali [6], gdzie po każdym teście zmieniano projekt sali według sugestii testujących, by uzyskać projekt dopasowany idealnie do potrzeb załogi szpitala.

W latach 2014-2015 L. Berg oraz J. Vance przeprowadzili szerokie badania dotyczące wykorzystania VR w przemyśle [7]. Skontaktowano i odwiedzono 18 firm zajmujących się między innymi motoryzacją, energetyką i rolnictwem.

Na miejscu okazało się, że firmy wykorzystują zarówno CAVE, HMD jak i połączenia tych technologii pozwalające na korzystanie z nich więcej niż jednej osobie na raz. Wirtualna rzeczywistość była wykorzystywana do procesu projektowania, a także rozwiązywania problemów locie. Najprzydatniejszym wykorzystaniem było badanie zakresu widzenia osób korzystających z projektowanego urządzenia w przypadkach takich jak zakres widzenia kierowcy z wnętrza samochodu, a także przeprojektowywania położenia urządzeń wewnątrz pojazdu. Przykładowo ułożenia klamek tak, by były one łatwo dostępne. Z pomocą VR prezentowane są również scenariusze. Przedstawienie zdarzeń w trójwymiarowej scenie działa na użytkowników znacznie lepiej niż pokaz slajdów. Łatwiej odnajdują się w sytuacji i szybciej pojmują prezentowane rozwiązania. Firmy stwierdziły również, że wykorzystanie wirtualnej rzeczywistości zmniejszyło koszty produkcji. Autorzy badania twierdzą, że VR nie jest rewolucją, a jednym z wielu przydatnych narzędzi w repertuarze firmy.

Nieustający rozwój technologii pozwala na zwiększenia rozdzielczości, co pozwoliłoby na zmniejszenie screen-door effect. Jest to efekt wynikający z odległości między oczami użytkownika a ekranem. Z tego względu widoczna jest siatka odstępów pomiędzy poszczególnymi pikselami. Drugą zaletą postępu technologicznego jest zmniejszenie wagi urządzeń, co znacząco wpływa na komfort ich użytkowania. Dodatkowo pojawiły się rozwiązania room-scale, czyli możliwość śledzenia ruchów użytkownika przy pomocy dodatkowych czujników [8], co umożliwia odwzorowywanie poruszania się użytkownika w świecie wirtualnym.

2. Cel i obiekt badań

Celem badania jest ocena możliwości i ograniczeń wirtualnej rzeczywistości w porównaniu z dostępnymi metodami prezentacji architektury, a także ocena przydatności tego rozwiązania na platformach mobilnych. Przed rozpoczęciem badania przedstawione zostały dwie hipotezy:

- Możliwe jest wykorzystanie skanów 3D i/lub projektów z aplikacji do projektowania do wizualizacji architektury na smartfonach.
- Wydajność smartfonu pozwala na wykorzystanie w takiej wizualizacji oświetlenia w czasie rzeczywistym.

W celu sprawdzenia tych hipotez przygotowana została wizualizacja w Unity 3D pozwalająca na poruszanie się po wirtualnym muzeum. Użytkownik może dowolnie obracać eksponaty i korzystać z interaktywnych tablic, by czytać ich opisy. Dodatkowo jeden z eksponatów można rozbić na mniejsze elementy.

Poza aplikacją wymagane są okulary. Hełm wirtualnej rzeczywistości w przypadku technologii mobilnych składa się najczęściej jedynie z obudowy, do której wkładany jest smartfon i soczewek skupiających [9]. W trakcie testów aplikacji wykorzystano kilka par okularów wirtualnej rzeczywistości.

Pierwsza para to Google Cardboard v1, czyli najprostsze i najtańsze dostępne na rynku rozwiązanie. Okulary te składają się z kartonu, przycisku magnetycznego i dwóch soczewek. Jest to minimum umożliwiające doświadczenie VR. Okulary te doczekały się następcy - Google Cardboard v2 (Rys. 1.). Jest to ulepszona wersja wcześniej opisanego modelu. Jest łatwiejsza do złożenia, posiada większe soczewki, współpracuje z większymi telefonami. Usprawniono również działanie przycisku. W tej edycji przycisk fizycznie naciska na ekran telefonu. Zapewnia to działanie przycisku za każdym razem, co nie było prawdą w przypadku v1.

Rys. 1. Okulary Google Cardboard v2. Srebrny przycisk widoczny po lewej stronie dociska srebrny pasek pomiędzy soczewkami do ekranu. Emuluje to dotknięcie ekranu palcem i umożliwia interakcję z aplikacją.

Bardziej złożonym rozwiązaniem są na przykład okulary Fiit VR 2n (Rys. 2.). Oferują one gąbkę umożliwiającą wygodniejsze ułożenie okularów na twarzy, miejsce pozwalające na włożenie okularów VR razem z okularami korekcyjnymi, a także suwaki umożliwiające zablokowanie telefonu w jednej pozycji. Co najważniejsze dostępna jest regulacja rozstawu soczewek, co pozwala na dopasowanie ich do konkretnej i wyeliminowanie problemów ze zniekształconym lub niewyraźnym obrazem. Możliwa jest również regulacja odległości soczewek od oczu, co pozwala na zrezygnowanie z okularów korekcyjnych na rzecz korekcji w okularach. Model ten nie posiada jednak przycisków.

Rys. 2. Okulary FiiT VR.

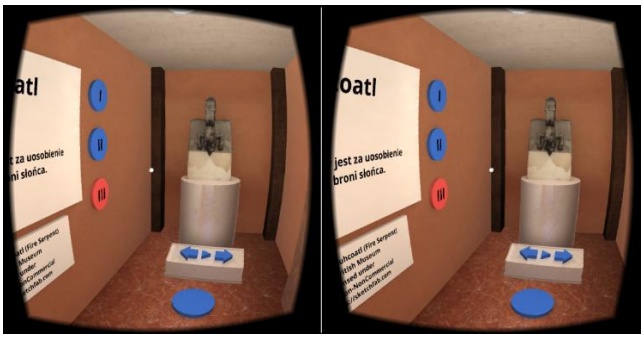
3. Przygotowanie wizualizacji

Wizualizacja została przygotowana na urządzenia mobilne z systemem operacyjnym Android. Został on wybrany ze względu na wsparcie okularów wirtualnej rzeczywistości oraz udostępnione przez Google VR SDK dla środowiska Unity. Do testów użyto smartfonu Xiaomi Redmi Note 3 (specyfikacja w Tabeli 1).

Tabela 1. Specyfikacja telefonu Xiaomi Redmi Note 3.

Procesor	Snapdragon 650 6x 1.8 GHz
Pamięć RAM	2 GB
Rozdzielczość ekranu	1920x1080 px
Wersja systemu Android	6.0.1

Google VR SDK (w wersji 1.0.3.) jest zestawem służących do integracji aplikacji przygotowanej w Unity z urządzeniami mobilnymi. Najważniejszym z nich jest gotowy prefab obsługi kamery GvrViewerMain. Generuje kamery dla lewego i prawego oka oraz dodaje obsługę ruchów głowy użytkownika (Rys. 3.).



Rys. 3. Działanie skryptu generującego oddzielny obraz dla każdego oka.

Aby ułatwić interakcję przed oczami użytkownika znajduje się wskaźnik (jest on częścią Google VR SDK). Pomaga on w skupieniu wzroku na wybranym przez niego obiekcie i ułatwia orientację w przestrzeni. Gdy wzrok nie jest skierowany na obiekt, z którym użytkownik może wejść w interakcję, wskaźnik pozostaje jednobarwnym okręgiem. Jeżeli przejdzie on na obiekt interaktywny zamienia się on w obręcz. Ze względu na charakterystykę wirtualnej rzeczywistości wszystkie elementy graficznego interfejsu użytkownika odwzorowane są jako trójwymiarowe obiekty w scenie.

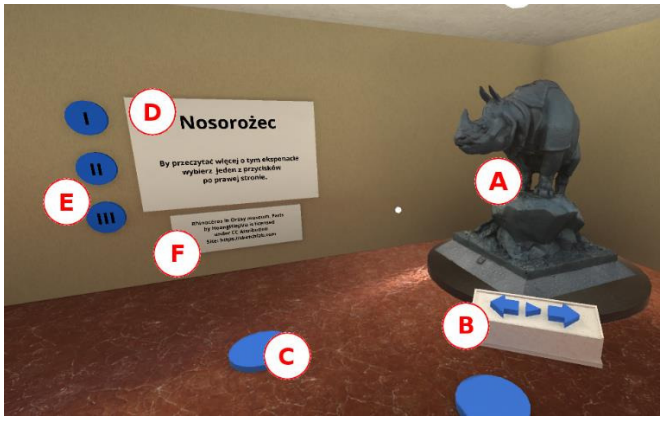
W ten sam sposób rozwiązana została interakcja z punktami, między którymi może poruszać się gracz. Mają one postać niebieskich, płaskich dysków. Gracz może je wskazać i, po wypełnieniu się dysku czerwony kolorem, przemieścić się w tę pozycję. Rysunek 4 obrazuje zachowanie analogiczne do przycisków menu. Od lewej strony jest to kolejno:

- dysk nie jest wskazywany przez gracza, zachowuje on niebieski kolor,
- gracz wskazał dysk wskaźnikiem, wskaźnik zamienił się w okrąg, nastąpiło rozpoczęcie odliczania zobrazowane rozszerzającym się, czerwonym okręgiem,
- dysk wypełnił się czerwonym kolorem, gracz został przemieszczony na jego pozycję,
- w sytuacji, gdy gracz przestał wskazywać na dysk, wypełnienie zaczyna się zmniejszać. Użytkownik ma możliwość wznowienia ładowania wskazując ponownie na dysk.

Rys. 4. Zachowanie punktów pod wpływem wskaźnika.

W każdej scenie umieszczony został licznik klatek - jego zadaniem będzie sprawdzenie wydajności aplikacji na urządzeniu mobilnym.

Do przygotowania planu muzeum i rozkładu rzeźb użyte zostały proste bryły udostępniane przez Unity. Z ich pomocą w krótkim czasie utworzony został mockup pomieszczeń. Głównym jego zadaniem było zaprojektowanie układu eksponatów pozostawiającego dużą przestrzeń na elementy sterujące oraz punkty, między którymi będzie mógł poruszać się gracz. Ich liczba powinna być jak największa, aby umożliwić graczowi jak największą swobodę poruszania się po pomieszczeniu. Po uzyskaniu satysfakcjonujących wyników układ pomieszczeń został wymodelowany w Blenderze, zaimportowany do Unity i wypełniony eksponatami.



Rys. 5. Elementy sceny.

Na scenę składają się elementy przedstawione na Rys. 5. Są to kolejno:

- **A - eksponat** - uproszczony model skanu 3D umieszczony na okrągłej podstawie,
- **B - panel** - pozwala na interakcję z rzeźbą - lewa i prawa strzałka obraca rzeźbę, dopóki wzrok pozostaje na przycisku. Przycisk w kształcie trójkąta pozwala na włączenie lub wyłączenie ciągłego obrotu,
- **C - teleport** - punkt pozwalający na przemieszczanie się użytkownika po scenie,
- **D - opis eksponatu** - tablica, na której wyświetlany jest tekst opisujący umieszczony obok eksponat, domyślnie nazwa eksponatu oraz tekst zachęcający do wybrania jednego z przycisków (oznaczonych na rysunku znakiem **E**),
- **E - przyciski** - pozwalają na zmianę tekstu wyświetlanego na elemencie **D** - użytkownik może się między nimi dowolnie przełączać przy pomocy wskaźnika,
- **F - licencja** - zawiera wszystkie informacje zawarte w wymaganiach licencji - nazwę autora, rodzaj licencji oraz stronę, z której pobrany został model.

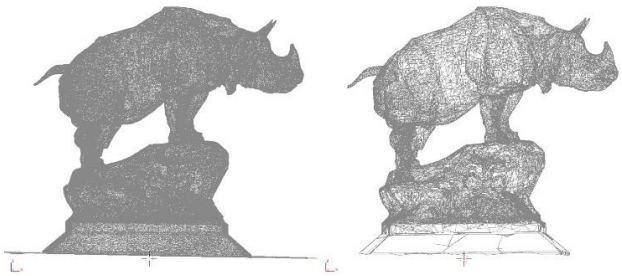
Wszystkie modele zeskanowanych rzeźb umieszczonych w scenie zostały pobrane ze strony Sketchfab. Udostępniane są one na licencji Creative Commons. Przed dodaniem ich do sceny wymagana była ich dodatkowa obróbka. Modele te charakteryzują się siatką złożoną z dużej liczby wierzchołków (od kilkudziesięciu do nawet ponad miliona). Ze względu na sprzęt obsługujący aplikację, smartfon, nie jest możliwe uzyskanie wrażenia płynności w aplikacji.

W związku z tym wymagane było znaczne uproszczenie siatki tych modeli.

Operacje związane z optymalizacją eksponatów wykonane zostały w programie graficznym Blender. Funkcja Decimate>Collapse pozwala ustalić procentową zawartość wielokątów w uproszczonym modelu względem modelu oryginalnego. Zastosowane zostało wielokrotne użycie tej funkcji z ustawieniami wartości w granicach 60-80% aż do uzyskania siatki składającej się z 20-30 tysięcy wierzchołków. Liczba wierzchołków została ustalona w trakcie badania opisanego w dalszej części artykułu. Niech za przykład posłuży statua nosorożca z Muzeum Orsay. Różnice w liczbie wierzchołków oraz ścian modelu przedstawiono w Tabeli 2, a podgląd siatki widoczny jest na Rys. 6.

Tabela 2. Porównanie modelu oryginalnego i uproszczonego.

	Model oryginalny	**Model uproszczony**
Liczba wierzchołków	175783	30703
Liczba trójkątów	351224	61607



Rys. 6. Porównanie siatki obu modeli. Model oryginalny (po lewej) oraz uproszczony (po prawej).

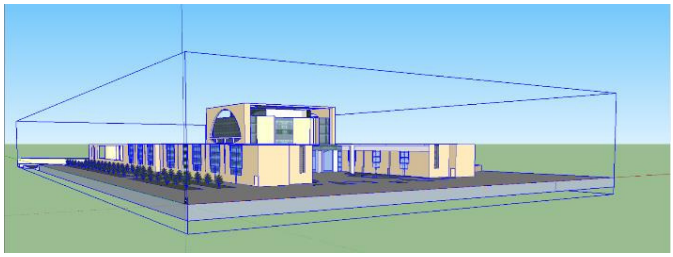
Testy wydajności przeprowadzone zostały przy pomocy aplikacji Trepn Profiler oraz przygotowanej wcześniej w Unity sceny wirtualnego muzeum. Badane miary to:

- maksymalna liczba generowanych klatek na sekundę,
- minimalna liczba generowanych klatek na sekundę,
- średnia liczba generowanych klatek na sekundę,
- średnie zużycie pamięci i zużycie pamięci w kolejnych sekundach testu,
- średnie obciążenie CPU i obciążenie CPU w kolejnych sekundach testu,
- średnie obciążenie GPU i obciążenie GPU w kolejnych sekundach testu.

Dodatkowo wykonano przegląd wybranych aplikacji umożliwiających eksport projektów jako modeli 3D. Przegląd ten miał za zadanie sprawdzenie możliwości prostego wykorzystywania przygotowanych wcześniej projektów w aplikacjach wirtualnej rzeczywistości. Umożliwia to przedstawienie modeli użytkownikom, którzy mogą zwiedzić budynek z perspektywy człowieka przechadzającego się po nim. Stwarza to niesamowite możliwości przedstawiania projektów jak i sprawdzania ich przydatności.

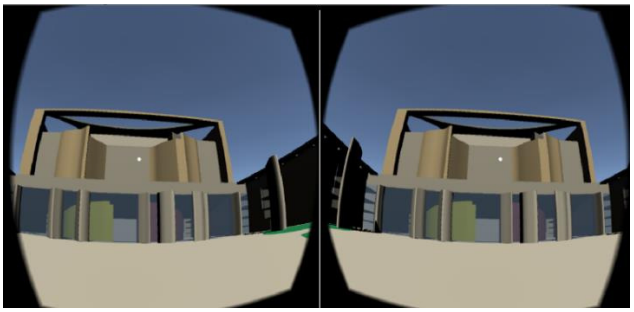
Do testów wykorzystanie zostały dwie dostępne za darmo aplikacje do projektowania architektury: Sketchup oraz Sweet Home 3D. Poza nimi na rynku dostępne jest wiele innych rozwiązań, zarówno darmowych, jak i płatnych. Wybór aplikacji powyżej jest czysto subiektywny, mający na celu przedstawienie możliwości eksportu gotowych modeli do aplikacji wirtualnej rzeczywistości.

Pierwszy z programów, Sketchup, udostępnia dodatkowo bibliotekę udostępnionych za darmo modeli. W związku z tym do eksportu przygotowany został jeden z dostępnych w bibliotece modeli - urząd kanclerza federalnego Niemiec (Rys. 7).



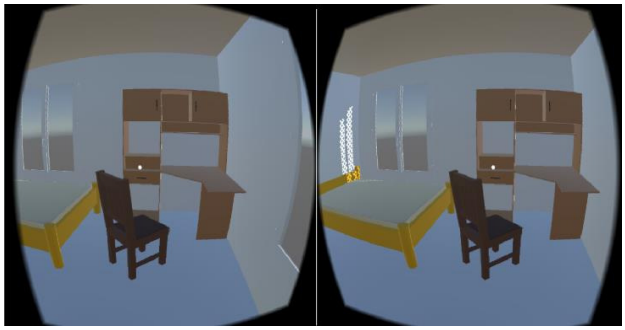
Rys. 7. Projekt w programie SketchUp.

Następnie został on wyeksportowany z programu SketchUp do formatu Collada (rozszerzenie .dae). Po bezpośrednim imporcie do Unity dostępny jest model z gotowymi już materiałami, które można modyfikować bezpośrednio wewnątrz środowiska. Po dokonaniu kilku poprawek materiałów dodano kamerę i elementy Google SDK. W ten sposób otrzymana została scena widoczna na Rys. 8.



Rys. 8. Projekt zaimportowany z programu SketchUp do sceny w Unity.

Drugi z programów, Sweet Home 3D, pozwala na planowanie pokojów przy pomocy prostych narzędzi i metody przeciągnij i upuść. Plan pokoju generowany jest jako podgląd 3D w czasie rzeczywistym. Program oferuje wyeksportowanie tak przygotowanego mieszkania do formatu .obj i następnie zaimportowanie go do Unity (Rys. 9).



Rys. 9. Projekt zaimportowany z programu Sweet Home 3D do sceny w Unity.

Analogiczny proces odbywa się w innych aplikacjach do projektowania. W ten sam sposób możliwe jest również importowanie modeli z programów takich jak AutoCAD, czy FreeCAD. Każdy z programów umożliwiających eksport do formatów takich jak .obj, czy .fbx umożliwia umieszczenie tych obiektów wewnątrz wirtualnej rzeczywistości i obejrzenie ich w trójwymiarze.

4. Przebieg testów

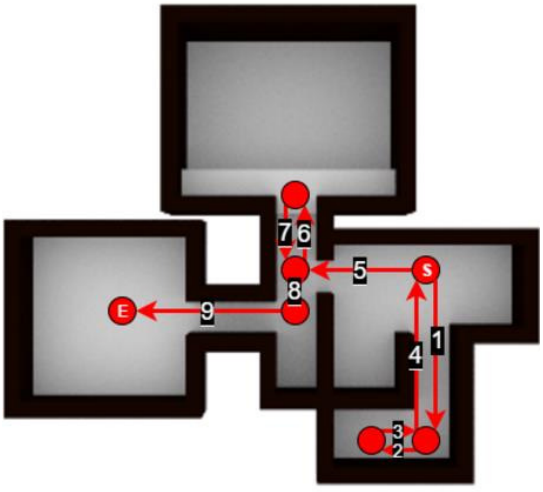
W trakcie testów wykorzystane zostały dwie metody badawcze:

- eksperyment na bazie jednego z eksponatów mający na celu dobór maksymalnej liczby wierzchołków umożliwiających płynne działanie aplikacji,
- metoda porównawcza, której zadaniem było porównanie wydajności sceny z oświetleniem obliczanym w czasie rzeczywistym ze sceną wykorzystującą lightmapy.

Pierwszy eksperyment opracowano na bazie modelu jednego z eksponatów. Przygotowana w Unity scena składała się z kilku kopii tego modelu oświetlonego czterema lampami punktowymi. Następnie model ten był stopniowo upraszczany aż do uzyskania zadowalającej wydajności (minimum 24 klatki na sekundę).

Zadaniem metody porównawczej było sprawdzenie wydajności dla dwóch wersji gotowej już aplikacji. Pierwsza, oświetlenie w czasie rzeczywistym, pozwala na uzyskanie dynamicznego światła i cieni reagujących na zmianę położenia obiektów w scenie. Druga korzysta z lightmap, które można wygenerować w trakcie tworzenia aplikacji, co pozwala na zwiększenie wydajności aplikacji.

Kolejne kroki wykonywane podczas porównywania dwóch scen to:



Rys. 10. Kolejne kroki podczas przeprowadzania testu w muzeum.

- wczytanie aplikacji,
- wybranie muzeum z listy scen,
- odwrócenie się w prawo o 180 stopni (S),
- przemieszczenie się w kierunku interaktywnej rzeźby (1, 2),
- rzucenie rzeźbą i obserwowanie jej rozbicia,
- powrót do punktu startowego (3, 4),
- przejście do pokoju z pomnikiem, obrót rzeźby (5, 6),
- przejście do pokoju z czterema rzeźbami (7, 8, 9),
- obrót o 360 stopni,
- zakończenie badania ze wzrokiem skierowanym na tablicę wyświetlającą liczbę wyświetlanych klatek na sekundę (E).

Numery w nawiasach obrazują przemieszczanie się wzdłuż ponumerowanych strzałek zaprezentowanych na Rys. 10.

5. Wyniki testów wydajności

Wyniki wydajności dla poszczególnych szczegółowości modeli zaprezentowane zostały w Tabeli 3.

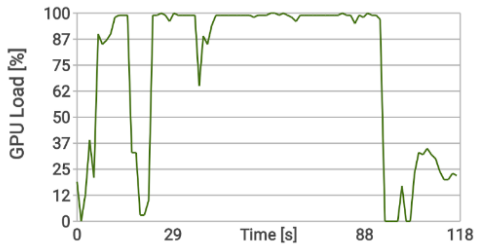
Tabela 3. Wydajność dla poszczególnych wersji modelu.

	Liczba wierzchołków	**Liczba klatek na sekundę**
Wersja A	190691	15
Wersja B	101420	17
Wersja C	62877	21
Wersja D	36524	24
Wersja E	22000	27
Wersja F	11807	30

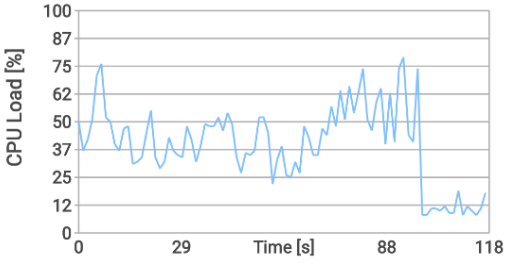
Wyniki drugiego testu przedstawione zostały w Tabeli 4. Kolumna Real-time opisuje wyniki dla sceny z oświetleniem w czasie rzeczywistym, a kolumna Lightmaps scenę z oświetleniem generowanym z użyciem lightmap. Dla drugiej sceny dodatkowo zaprezentowane zostało procentowe obciążenie GPU (Rys. 11.) i CPU (Rys. 12.).

Tabela 4. Wydajność dla wizualizacji z oświetleniem w czasie rzeczywistym oraz oświetleniem przy pomocy lightmap.

	Real-time	**Lightmaps**
Max FPS	28	48
Min FPS	4	28
Średnie FPS	16	40
Średnie obciążenie CPU [%]	41,28	39,04
Średnie obciążenie GPU [%]	77,31	81,92
Średnie zużycie pamięci [MB]	1855,8	1831,33



Rys. 11. Obciążenie GPU w trakcie działania aplikacji (lightmapy).



Rys. 12. Obciążenie CPU w trakcie działania aplikacji (lightmapy).

6. Opis wyników i wnioski

Wstępne testy przedstawione w Tabeli 3 pozwoliły na ocenę możliwości urządzenia i ustalenie zakresu liczby wierzchołków eksponatów. Do płynnego działania liczba wierzchołków nie powinna przekraczać 36 tysięcy dla testowanego modelu telefonu.

Następny test miał na celu porównanie wydajności aplikacji z oświetleniem generowanym w czasie rzeczywistym i aplikacji z wypalonymi lightmapami. Jak widać w Tabeli 4 oświetlenie w czasie rzeczywistym nie pozwala na osiągnięcie minimalnej liczby klatek wymaganej do płynnego wyświetlania obrazu - jest to średnio 16 klatek na sekundę. Dopiero przygotowanie lightmap pozwoliło na osiągnięcie zadowalających wyników. Średnio 40 klatek z minimalną liczbą klatek na poziomie 28. Jest to zadowalający wynik pozwalający na bezproblemową interakcję z aplikacją. W przypadku CPU obciążenie nigdy nie przekroczyło wartości 75% (Rys. 12.). Widocznym słabym punktem było jednak GPU. Często osiągało obciążenie na poziomie 100% (Rys. 11.). W związku z tym druga z hipotez postawiona przed rozpoczęciem badań jest nieprawdziwa. Wydajność testowanego urządzenia była niewystarczająca do wykorzystania w scenie z oświetleniem w czasie rzeczywistym.

Zarówno wykorzystanie skanów 3D jak i aplikacji do projektowania znacząco skraca proces tworzenia wizualizacji architektonicznych, nawet biorąc pod uwagę wymaganą wstępną obróbkę skanów. Potwierdza to pierwszą hipotezę postawioną przed rozpoczęciem badania. Dostępne na rynku rozwiązania pozwalają na relatywnie prostą obróbkę modeli do wykorzystania w aplikacji mobilnych. Pozwala to na przygotowania wizualizacji architektury bez posiadania umiejętności w zakresie modelowania 3D. Uzyskane w ten sposób obiekty pozwalają na odwzorowanie rzeczywistości w najdrobniejszych szczegółach. Największą przewagą modeli przygotowanych ręcznie jest ich optymalizacja. W przypadku skanów 3D zaobserwowana dużą liczbę wierzchołków nie wpływających znacząco na szczegółowość modelu, które mogą stanowić problem w trakcie procesu automatycznego upraszczania siatki modelu.

Wykorzystywane w badaniu okulary Cardboard umożliwiały komfortowy odbiór, ale brak konfiguracji rozstawu soczewek, dostępny w okularach FiiT VR, znacząco utrudniał uzyskanie wyraźnego obrazu jeżeli z okularów korzystała więcej niż jedna osoba. Udostępnione przez Google narzędzia do konfiguracji pozwalają na przygotowanie profilu dedykowanego dla poszczególnych użytkowników. W przypadku testowanego smartfonu o rozdzielczości 1920x1080 pikseli nadal widoczny jest jednak screen-door effect. Rozwiązaniem tego problemu jest wykorzystanie smartfonu z większą rozdzielczością.

Okulary wpływają również na decyzje podejmowane podczas projektowania aplikacji. Kwestie takie jak sposób poruszania się oraz interakcja z otoczeniem, to problemy, które trzeba rozwiązać przed rozpoczęciem tworzenia wizualizacji. HMD dostępne na komputerach stacjonarnych posiadają kontrolery odwzorowujące ruchy rąk. W przypadku rozwiązań mobilnych niektóre modele okularów nie posiadają fizycznych przycisków, lub jest to tylko jeden przycisk. Możliwe jest użycie bezprzewodowych kontrolerów bluetooth, co może jednak ograniczyć liczbę użytkowników posiadających możliwość interakcji z aplikacją. Dlatego też w przypadku testowanej aplikacji postawiono na rozwiązania umożliwiających korzystanie z aplikacji na jak najszerszej liczbie urządzeń. Kontrola przy pomocy skupiania wzroku pozwala na korzystanie z aplikacji bez użycia fizycznych kontrolerów. Wybrany sposób poruszania się to najpopularniejsza obecnie metoda, czyli poruszanie się pomiędzy punktami. Jest to jednak rozwiązanie mało komfortowe i badane są obecnie inne rozwiązania.

Podsumowując, odwzorowanie architektury w wirtualnej rzeczywistości na urządzeniach mobilnych jest jak najbardziej możliwe i ma bardzo duży potencjał. Wykorzystanie w niej skomplikowanych modeli wymaga jednak większej wydajności procesorów graficznych w smartfonach, szczególnie tych z niższej półki, które zostały wykorzystywane do testów przedstawionych w artykule. Obserwując ciągły rozwój zarówno smartfonów jak i okularów wirtualnej rzeczywistości uzyskanie satysfakcjonującej jakości obrazu jest tylko kwestią czasu.

Literatura

[1] „The CAVE: audio visual experience automatic virtual environment," Communications of the ACM, pp. 64-72., Nr 6/35.

[2] „A review Paper on Oculus Rift – A Virtual Reality Headset," International Journal of Engineering Trends and Technology, pp. 175-179., 2014.

[3] M. Fuchs, „PlastiCity A Multiplayer Urban Planning Game," Space Time Play Computer games, architecture and urbanism: the next level, pp. 370-371, 2007.

[4] B. J. Fernández-Palacios, D. Morabito i F. Remondino, „Access to complex reality-based 3D models using virtual reality solutions," Journal of Cultural Heritage, pp. 40-48, 23/2017.

[5] „Virtual and Augmented Reality for Rich Interaction with Cultural Heritage Sites: A Case Study from the Roman Theater at Byblos," Digital Applications in Archaeology and Cultural Heritage, 2017.

[6] J. Konopacki, „The technology of augmented reality - virtual reconstructions of landscape architecture design," Technical Transactions, nr 5-A, pp. 98-112, 2014.

[7] A. Ibrahim, J. Dimick i A. Joseph, „Building a Better Operating Room," Annals of Surgery, p. 35, 1/2017.

[8] L. Berg i J. Vance, „Industry use of virtual reality in product design and manufacturing: a survey," Virtual Reality, pp. 1-17, 21/2017.

[9] S. Greenwald, Z. Wang, M. Funk i P. Maes, „Investigating Social Presence and Communication with Embodied Avatars in Room-Scale Virtual Reality," w Immersive Learning Network Conference, 2017.

[10] Z. Deng, „Design and fabrication of concave-convex lens for head mounted virtual reality 3D glasses," w International Conference on Optical Instruments and Technology, 2015.

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 193-199
Wysłane: 2017-06-27
Przyjęte: 2017-06-29

Analiza możliwości wykorzystania czujników urządzeń mobilnych pod kontrolą zmodyfikowanych systemów operacyjnych

Mateusz Dobrowolski*, Michał Dobrowolski *, Piotr Kopniak*
[a] Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W niniejszej publikacji poruszono temat możliwości wykorzystania czujników urządzeń mobilnych pod kontrolą zmodyfikowanych systemów operacyjnych. Przeprowadzone badania dotyczą urządzeń z systemem Android. Do odczytów danych z żyroskopu, akcelerometru, orientacji oraz miernika światła posłużyła darmowa aplikacja Physics Toolbox Sensor. Badania zostały przeprowadzone na dwóch urządzeniach producenta telefonów Xiaomi pod kontrolą sześciu różnych systemów. Pomiary czujników zweryfikowano za pomocą dokładnych czujników referencyjnych.

Słowa kluczowe: urządzenia mobilne, czujniki, system Android

*Autor do korespondencji.

Adres/adresy e-mail: mateusz.dobrowolski13@gmail.com, michal.dobrowolski000@gmail.com,p.kopniak@pollub.edu.pl

Analysis of the use of sensors in mobile devices with modified operating systems

Mateusz Dobrowolski*, Michał Dobrowolski *, Piotr Kopniak*
[a] Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This publication concentrate on the posibility of the use of sensors in mobile devices with modified operating systems. Presented research focuses on Android devices. The gyroscope, the accelerometer, the orientation sensor and the light sensor data was acquired with use of Physics Toolbox Sensor software. The research has been conducted on two mobile devices of Xiaomi under control of six different kinds of operating system. Measured values were compared to values recorded by very accurate, reference sensors.

Keywords: mobile devices, sensors, Android

*Corresponding author.

E-mail address/addresses: mateusz.dobrowolski13@gmail.com, michal.dobrowolski000@gmail.com,p.kopniak@pollub.edu.pl

1. Wstęp

W dzisiejszych czasach smartfony odgrywają bardzo dużą rolę. Niemal każda osoba wyposażona jest w urządzenie mobilne. Na skutek globalnego rozwoju informatyzacji, stały się one integralną częścią życia - zarówno osobistego, jak i biznesowego. Popularne smatrfony mogą pracować pod kontrolą wielu różnych systemów operacyjnych.

Na obecnym runku najpopularniejszym systemem jest Android, cieszący się dużym zainteresowaniem wśród odbiorców ze względu na prostotę użytkowania i duże wsparcie techniczne [1]. Jest platformą o otwartym kodzie, dzięki czemu programiści mogą modyfikować kod poprzez dodawanie nowych funkcji, personalizacji całego wyglądu systemu do swoich upodobań oraz optymalizacji działań mających na celu energooszczędną pracę urządzenia mobilnego. Dlatego powstaje wiele odmian tego systemu proponowanych przez firmy zewnętrzne i różne społeczności.

Dzięki szybkiemu rozwojowi sprzętu i miniaturyzacji sprzętu, urządzenia mobilne wyposażone są w dużą liczbę czujników, głównie inercyjnych.. Do najpopularniejszych z nich i najczęściej stosowanych w urządzeniach możemy wymienić między innymi. akcelerometr, żyroskop i magnetometr. Wydajność czujników różni się znacznie w zależności od modelu smartfonów. Jednym ze szczególnych

Kolejnym przypuszczeniem jest to, że sensory urządzenia mobilnego charakteryzują się niską precyzją i dokładnością zastosowań czujników inercyjnych jest śledzenie ruchu w różnych systemach mobilnych i aplikacjach

Tematyka analizy możliwości wykorzystania czujników telefonów nie jest tematem nowym, a dostępne artykuły badawcze dogłębnie opisują poruszane zagadnienia. Pierwszy artykuł dotyczy możliwości wykorzystania czujników inercyjnych urządzenia mobilnego dla inteligentnych systemów transportowych. Badania sensorów zostały przeprowadzone pod kontrolą systemu Android oraz iOS [2]. Drugi artykuł dotyczył badania możliwości wykorzystania akcelerometrów do rejestrowania przyśpieszenia. Autorzy badali czujnik przyśpieszenia pod kątem pracy na trzech platformach mobilnych: Android, Windows Phone oraz iOS [3]. Ostatnia praca dotyczyła ocenę wydajności sensorów urządzeń mobilnych w zakresie ruchu. Celem badań była ocena pracy czujników i wydajności algorytmów wykorzystywanych do pomiarów kątów pod kontrolą systemu Android i iOS [4].

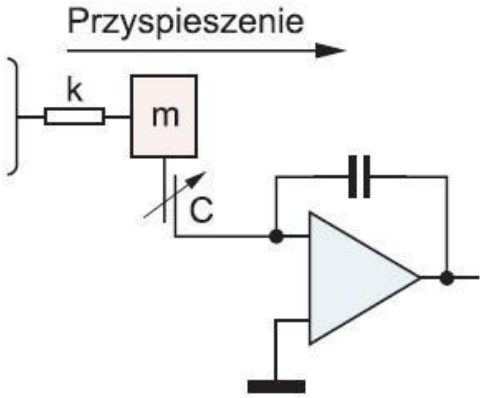
Celem opisanej w niniejszym artykule pracy badawczej było udowodnienie, że dokładność pomiarów wielkości fizycznych za pomocą czujników wbudowanych w urządzenia mobilne jest uzależniona od zastosowanego systemu operacyjnego. Oczywiste może wydawać się, że największą dokładność mają czujniki pod kontrolą fabrycznego systemu operacyjnego, ale wyniki badań pokazują, że nie jest to regułą. pomiarową co też należało zweryfikować wykorzystując referencyjne urządzenia pomiarowe.

2. Charakterystyka sensorów

Dynamiczny rozwój technologii elektronicznej oraz wzrost popularności smartfonów spowodowały, że czujniki takie jak akcelerometr, żyroskop, magnetometr czy odbiornik GPS, które niegdyś uważane były za specjalistyczne narzędzia, stały się dostępne dla każdego użytkownika urządzeń mobilnych. Telefony wyposażone w te sensory stały się bogatsze o nowe funkcjonalności. Dzięki nim korzystanie z urządzeń mobilnych staje się coraz prostsze i wygodniejsze. Badaniom w niniejszej pracy poddane zostały czujniki występujące w każdym współczesnym smartfonie, tzn. czujnik pomiaru natężenia światła, akcelerometr, żyroskop i magnetometr. W niniejszym rozdziale pokrótce zostanie scharakteryzowany każdy z czujników.

2.1. Akcelerometr

Akcelerometr jest to narzędzie służące do pomiaru przyśpieszenia liniowego lub kątowego urządzeń w niego wyposażonego. Smartfony oraz inne urządzenia mobilne najczęściej posiadają akcelerometr liniowy. Mierzy on przyśpieszenie wzdłuż każdej z osi trójwymiarowego układu współrzędnych, czyli: X, Y i Z. W jego wnętrzu (Rys. 1) znajduje się bezwładny odważnik, który pod wpływem ruchu przekazuje odpowiednie sygnały do silniczków, które przeciwdziałają temu ruchowi. Wartości przyśpieszenia, zatem zmieniają swoją wartość wraz ze wzrostem lub spadkiem szybkości oraz zmianą kierunku poruszania urządzenia. W przypadku, gdy sprzęt zostaje w stanie spoczynku, czujnik wskazuje wartości zerowe dla każdego z wymiarów.



Rys. 1. Schemat budowy akcelerometru [5]

Głównym elementem czujnika jest tzw. masa bezwładna, zamocowana na sprężystych belkach, która stanowi elektrodę w układzie kondensatora pomiarowego. W wyniku przemieszczenia masy zmienia się pojemność i tym samym napięcie wyjściowe, które jest dalej przetwarzane np. w przetworniku A/C i mikrokontrolerze [6]. Akcelerometr w urządzeniach mobilnych często wykorzystywany jest w szeregu aplikacjach czy do stabilizacji obrazu z obiektywu aparatu.

2.2. Żyroskop

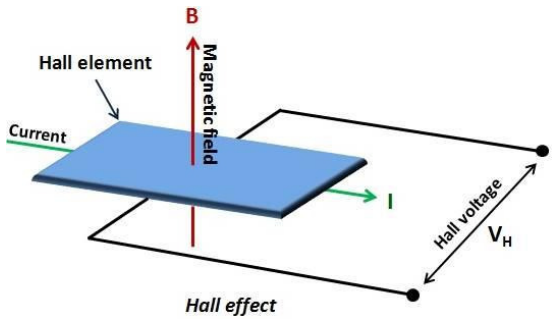
Żyroskop to czujnik służący do pomiaru położenia kątowego urządzenia. Mierzy on prędkość kątową obracającego się urządzenia wokół osi X, Y lub Z. Zbudowany jest on z mikroskopijnych płytek o grzebieniowatej strukturze. Pod wpływem impulsu elektrycznego płytki te ulegają wibracjom. Na skutek obrotu urządzenia, płytki zmieniają kierunek drgań. Spowodowane jest to tak zwanym efektem Coriolisa, czyli odchylenia danego obiektu od jego toru ruchu widziane z perspektywy innych obiektów znajdujących się w wirującym układzie współrzędnych. W przypadku żyroskopu obiekty te to płytki, które w wyniku ruchu urządzenia mobilnego, zmieniają swój potencjał elektryczny. Zmiany te interpretowane są przez sensory jako prędkość obrotowa [6]. Przykładem zastosowania żyroskopu w urządzeniach mobilnych może być automatyczne obracanie ekranu czy w różnego rodzaju aplikacjach, które korzystają z wykrywania położenia telefonu.

2.3. Miernik światła

Zastosowanie tego czujnika w urządzeniach mobilnych jest kluczową zaletą. Sensor ten składa się z fotorezystora, który z kolei zawiera półprzewodniki zezwalające na przepływ prądu tylko w przypadku, gdy uderzające w niego fale mają dokładny zakres częstotliwości. Często miernik światła wyposażony jest w czujnik podczerwieni oraz filtr ultrafioletu. Dzięki nim ogranicza się rodzaj fal, które mogą osiągnąć półprzewodniki w widmie widzialnym światła. Czujnik ten działa identycznie dla różnych źródeł światła, począwszy od naturalnego, a kończąc na świetle ze świetlówek i żarówek. Wzrost natężenia prądu pod wpływem światła interpretowany jest jako wzrost natężenia światła. Jego główna zaleta to przede wszystkim adaptacja jasności wyświetlacza, odpowiedniej dla ludzkiego oka, poprzez właściwe regulowanie podświetlenia wyświetlacza LCD. To działanie pozwala na zmniejszenie zmęczenia oczu użytkownika [7].

2.4. Magnetometr

Magnetometr to czujnik wykrywający pole magnetyczne Ziemi wzdłuż trzech prostopadłych do siebie płaszczyznach X, Y oraz Z. Zasada działania magnetometru w urządzeniach mobilnych opiera się o tzw. metodę Halla (Rys. 2). Składa się on z płaskiej płytki, która przewodzi prąd elektryczny w każdym wymiarze. Czujnik ten wytwarza napięcie proporcjonalnie do siły i biegunowości pola magnetycznego wzdłuż każdego wymiaru, tym samym polaryzuje ją. Mierzona jest ona, jako różnica potencjału na krawędziach płytki Wzrost wartości potencjału powoduje zwiększenie natężenia pola magnetycznego w danym wymiarze. Zmierzone napięcie przekształcane jest do sygnału cyfrowego, które reprezentuje natężenie pola magnetycznego. Magnetometr w urządzeniach mobilnych wykorzystywany jest jako kompas. Jest on podatny jednak na zakłócenia elektromagnetyczne.



Rys. 2. Schemat budowy magnetometru [8]

Wyróżnia się również magnetometry, wykorzystujące efekt magnetooporowy. Składają się one z magnesów trwałych w obwodzie. Gdy magnes wystawiony jest na działanie zewnętrznych pól magnetycznych to zmienia się opór magnesu w obwodzie [9].

3. Konfiguracja układu badawczego

Tak jak wspomniano wcześniej, przedmiotem badań były czujniki urządzeń mobilnych: miernik światła, orientacji, akcelerometru i żyroskopu.

Do przeprowadzenia eksperymentów użyto dwóch urządzeń mobilnych z systemami Android. Producentem telefonów jest firma Xiaomi. Oba urządzenia domyślnie pracują pod kontrolą oficjalnego systemu chińskiego dystrybutora o nazwie MIUI. Jest to oprogramowanie, które zostało stworzone przez firmę Xiaomi dla smartfonów i tabletów, oferującego swobodę personalizacji telefonu pod potrzeby użytkownika. Oba urządzenia różnią się między sobą wielkością ekranu oraz modelem procesora[10]. Specyfikacja sensorów została odczytana z urządzeń i przedstawiona w tabeli 1.

3.1. Metody i przebieg badań

Do badań wybrano pięć systemów alternatywnych dla oryginalnego systemu Android.. Głównymi kryteriami wyboru systemu operacyjnego dla obu urządzeń mobilnych była zgodność rodzaju i wersji systemu. Lista wykorzystanych systemów została przedstawiona w tabeli 2

W przypadku badania miernika światła, punktem odniesienia był cyfrowy miernik światła EXTECH LT300 o dokładności pomiarowej przedstawionej w tabeli 3.

Dla czujnika orientacji, akcelerometru oraz żyroskopu punktem odniesienia do rzeczywistych pomiarów był rejestrator ruchu MTw Awinda, będącym integralną częścią systemu firmy Xsens. W tabeli 4 zaprezentowane zostały wybrane parametry urządzenia MTw.

Tabela 1. Parametry czujników urządzeń mobilnych [odczytane z urządzeń]

Cecha	Xiaomi Redmi Note 3 Pro	Xiaomi Redmi 3 Pro
Akcelerometr		
Producent	BOSH	Invensense
Wersja	2061000	1
Natężenie prądu	0.18 mA	0.5 mA
Rozdzielczość	0.0023956299 m/s2	0.0005981440 m/s2
Maksymalny zakres	39,2266082764 m/s2	156.8000030518 m/s2
Żyroskop		
Producent	BOSH	Invensense
Wersja	2061000	1
Natężenie prądu	0.9 mA	3.6 mA
Rozdzielczość	0.0010681152 rad/s	0.0010681152 rad/s
Maksymalny zakres	34,9065856934 rad/s	34,9065856934 rad/s
Magnetometr		
Producent	Yamaha	Yamaha
Wersja	35193090	1
Natężenie prądu	0.28 mA	0.28 mA
Rozdzielczość	0.3005981445 uT	1 uT
Maksymalny zakres	2000 uT	2000 uT
Sensor światła		
Producent	Liteon	Liteon
Wersja	1	1
Natężenie prądu	0.1 mA	0.2 mA
Rozdzielczość	0.0149993896 lx	0.0125000002 lx
Maksymalny zakres	10000 lx	60000 lx
Przyśpieszenie liniowe		
Producent	QTI	AOSP
Wersja	2	3
Natężenie prądu	1.0799866 mA	4.38 mA
Rozdzielczość	0.0023956299 m/s2	0.000598144 m/s2
Maksymalny zakres	39.2266082764 m/s2	19.6133003235 m/s2

Tabela 2. . Systemy operacyjne Android wykorzystane w badaniu

Nazwa systemu	Xiaomi Redmi Note 3 Pro	Xiaomi Redmi 3 Pro
LineageOS-14.1	X	X
RessoructionRemix N 5.8.1	X	X
MoKee 71.0	X	X
CyanogenMod 13.0	X	X
FlymeOS 5	X	X
MIUI 6.9.29	X	X

Tabela 3. Systemy operacyjne Android wykorzystane w badaniu

Nazwa systemu	Xiaomi Redmi Note 3 Pro	Xiaomi Redmi 3 Pro
LineageOS-14.1	X	X
RessoructionRemix N 5.8.1	X	X
MoKee 71.0	X	X
CyanogenMod 13.0	X	X
FlymeOS 5	X	X
MIUI 6.9.29	X	X

Tabela 4. Charakterystyka dokładności pomiarowej cyfrowego miernika światła EXTECH LT300 [11]

Zakres i rozdzielczość	Dokładność
Luksy	
40.00, 400.0, 4000, 40.00k, 400.0kLux	± (5% odczytu + 0.5% pełnej skali)
Stopoświece (Fc)	
40.00, 400.0, 4000, 40.00kFc	± (5% odczytu + 0.5% pełnej skali)

Tabela 5. Wybrane parametry inercyjnego czujnika MTw [12]

Nazwa cechy	Wartość
Dokładność statyczna (Roll / Pitch)	<0.5°
Dokładność dynamiczna	2.0°
Rozdzielczość kątowa	0.05°
Zakres temperatur pracy	-10°C – 60°C
Częstotliwość radia	ISM 2.4GHz
Zakres transmisji	Otwarta przestrzeń - do 50m Zamknięta przestrzeń – do 20m
Wewnętrzna częstotliwość próbkowania	1800Hz
Szybkość aktualizacji bezprzewodowej	1 MTw 120 Hz 6 MTw 75 Hz 12 MTw 50 Hz 32 MTw 20 Hz
Masa	27g

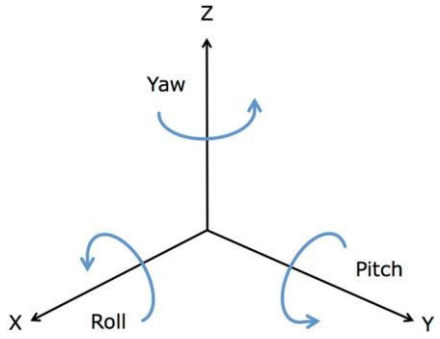
Pierwszym badanym czujnikiem w obu urządzeniach mobilnych pod kontrolą sześciu różnych systemów operacyjnych był miernik światła. Badania skupiły się na określeniu różnic w pomiarach uzyskanych przez miernik światła w urządzeniu mobilnym oraz porównaniu ich względem cyfrowego miernika. Proces rejestracji światła trwał 10 sekund przy najwyższej częstotliwości próbkowania wynoszącej 100Hz i został przeprowadzony w warunkach oświetleniowych:

- sztucznego światła, o natężeniu wynoszącym 3387 Lux, pochodzącego z świetlówki Philips o mocy 15W. Odległość od źródła do obiektów badawczych wynosiła 100cm.
- naturalnego oświetlenia w otwartej przestrzeni, o natężeniu wynoszącym 64.2 kLux.

Wszystkie pomiary za pomocą mierników światła zostały przeprowadzone w identycznych warunkach i w tym samym położeniu urządzeń badanych.

Kolejny eksperyment dotyczył czujnika orientacji. System Android oblicza kąt orientacji przy użyciu czujnika pola magnetycznego w połączeniu z akcelerometrem [9]. Dane zebrane z powyżej wymienionych czujników sprzętowych, dostarczają nam informacji o orientacji urządzenia dla następujących trzech osi (Rys. 3):

- Azymut
- Pochylenie
- Przechylenie



Rys. 3. Układ współrzędnych [XIII]

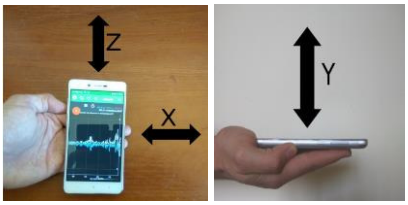
Wartości zarejestrowane przez czujniki urządzenia MTw Awinda stanowiły punkt wyjścia do określenia różnic wyników zarejestrowanych przez czujniki telefonów pod kontrolą zmodyfikowanych systemów. Badanie azymutu rozpoczynało się poprzez ułożenie urządzeń w taki sposób, aby ich osie Z były skierowane prostopadle do podłoża oraz osie Y wskazywały na przybliżony kierunek północy (Rys. 4c). Kolejnym etapem był ruch urządzeń, co 15° w prawą stronę, aż do wykonania pełnego obrotu (360°). W punktach kontrolnych spisywane były wartości wskazywane przez czujniki urządzeń mobilnych oraz urządzenia MTw Awinda. Kolejnym etapem było badanie kąta pochylenia, rozpoczynającego się od ustawienia urządzeń w taki sposób, aby urządzenia były położone prostopadle do powierzchni Ziemi (Rys. 4a). Następnym etapem był ruch urządzeń, co 15° wokół osi X, do momentu wykonania pełnego obrotu. Podobnie jak we wcześniejszym przypadku, w określonych punktach kontrolnych, z obu czujników rejestrowane były wartości wskazywane przez urządzenia. Badanie kąta przechylenia rozpoczynało się wykonaniem obrotu urządzeń mobilnych wraz czujnikiem MTw Awinda wokół osi Z, co 15° w prawo do momentu wykonania pełnego obrotu (Rys. 4b). Etap odczytu kąta przechylenia określonego przez czujnik orientacji urządzenia mobilnego przebiegał w taki sam sposób jak w poprzednich przypadkach.

Ostatnie badanie dotyczyło czujnika przyśpieszenia oraz żyroskopu. Polegało na sekwencji ruchowej (Rys. 5) urządzeń mobilnych wraz z czujnikiem MTw Awinda w trzech kierunkach wzdłuż osi X, Y i Z. Badanie osi X rozpoczynało się przez przesuwanie urządzeń w prawo na daną odległość i powrotu do pozycji wyjściowej. Badanie wzdłuż osi Y i Z, ruch odbywał się odpowiednio w kierunku danej osi. Sekwencja ruchu wykonywana była w dwóch tempach: wolnym i szybkim. Dla każdej z nich ruch został powtórzony pięciokrotnie w kierunku i naprzeciw zwrotu osi. Dla wszystkich czujników fizycznych została ustawiona najwyższa możliwa częstotliwość pobierania próbek wynosząca 100Hz. W ten sposób zapewniono dostarczenie

większej ilości danych, w celu zapewnienia większej dokładności podczas analizy działania akcelerometru i żyroskopu.

a) b) c)

Rys. 4. Sposób wykonywania ruchów podczas badania: a) pochylenia, b) przechylenia, c) azymutu



Rys. 5. Sposób wykonywania ruchów podczas badania akcelerometru w trzech kierunkach wzdłuż osi X, Y i Z

3.2. Wyniki badań

W niniejszym podrozdziale zostaną przedstawione wyniki przeprowadzonych badań. Tabela 5 zawiera procentowe różnice pomiędzy natężeniem światła zarejestrowanych przez czujniki urządzeń mobilnych, a miernikiem cyfrowym. Wyniki dotyczą pomiarów uzyskanych w warunkach naturalnego i sztucznego oświetlenia. W tabeli 6 przedstawiono średnie wartości odchylenia ze wszystkich punktów pomiarowych, które zarejestrowane zostały przez czujniki orientacji urządzeń mobilnych. W tabeli 7 zostały zawarte informacje na temat dokładności pomiarów przyśpieszeń liniowych, w trakcie całego przebiegu badania akcelerometru. Stopień dokładności został obliczony za pomocą współczynnika determinacji (1).

$$R^2 = \frac{\sum_{t=1}^{n}\left(\hat{y}_t - \bar{y}\right)^2}{\sum_{t=1}^{n}\left(y_t - \bar{y}\right)^2} \quad (1)$$

gdzie: $\hat{y}_t$ - wartość teoretyczna, $\bar{y}$ - średnia arytmetyczna empirycznych wartości zmiennej objaśnianej, y_t - rzeczywista wartość, R^2 współczynnik determinacji.

Tabele 8 i 9 przedstawiają maksymalne bezwzględne wartości przyśpieszeń liniowych. Pomiary uzyskane przez czujniki urządzeń mobilnych zostały porównane z odczytami MTw Awinda, na podstawie czego wyznaczono procentową różnicę pomiędzy nimi. Ostatnie tabele 10 i 11 zawierają maksymalne prędkości kątowe, zarejestrowane przez czujniki urządzeń mobilnych i MTw Awinda. Dolna część tabel stanowi informację na temat procentowej różnicy w uzyskanych pomiarach w odniesieniu do rzeczywistych wartości z sensora Xsens.

Tabela 6. Wyniki badań natężenia światła w warunkach sztucznego i naturalnego światła

System	Xiaomi Redmi Note 3 Pro		Xiaomi Redmi 3 Pro	
	Dzienne światło	Sztuczne oświetlenie	Dzienne światło	Sztuczne oświetlenie
MIUI	94,89%	79,47%	87,92%	6,01%
CyanogenMod	94,11%	76,77%	86,15%	7,57%
MoKee	94,47%	78,69%	87,17%	8,01%
Resurrection Remix OS	94,27%	80,92%	86,52%	7,56%
LineageOS	94,67%	81,78%	87,47%	10,26%
Flyme OS	94,47%	82,00%	86,18%	8,56%

Tabela 7. Wyniki badań czujnika orientacji

Ruch / System	Xiaomi Redmi Note 3 Pro			Xiaomi Redmi 3 Pro		
	Pochylenie [°]	Przechylenie [°]	Azymut [°]	Pochylenie [°]	Przechylenie [°]	Azymut [°]
MIUI	0,75	0,51	12,84	0,92	1,32	8,18
CyanogenMod	1,93	2,5	13,35	1,63	1,56	7,23
MoKee	1,84	2,6	9,97	1,62	1,43	7,07
Resurrection Remix OS	1,85	2,5	14,43	1,6	1,48	7,55
LineageOS	1,83	2,63	9,26	1,2	1,83	0,85
Flyme OS	0,6	0,35	14,84	1,76	1,3	7,06

Tabela 8. Wyniki badań dokładności działania akcelerometrów urządzeń mobilnych

	Xiaomi Redmi Note 3 Pro		Xiaomi Redmi 3 Pro	
System	**Sekwencja wolna**	**Sekwencja szybka**	**Sekwencja wolna**	**Sekwencja szybka**
MIUI	82,77%	66,85%	75,73%	79,55%
CyanogenMod	29,81%	66,09%	47,25%	63,37%
MoKee	45,59%	69,09%	55,73%	62,80%
Resurrection Remix OS	61,72%	43,39%	53,85%	48,10%
LineageOS	66,91%	67,95%	71,07%	73,03%
Flyme OS	68,99%	55,07%	51,01%	18,54%

Tabela 9. Wyniki badań maksymalnych bezwzględnych wartości przyśpieszeń liniowych, urządzenie Xiaomi Redmi 3 Pro

Oś	**System**	**MIUI**	**CyanogenMod**	**MoKee**	**Resurrection Remix OS**	**LineageOS**	**Flyme OS**
Xiaomi Redmi 3 Pro – wolna sekwencja ruchu [m/s2]							
X	**Telefon**	0,5201	0,7454	0,4678	0,6604	0,4727	0,5651
	MTw Awinda	1,3744	5,2573	2,7801	1,3844	2,0228	1,1481
Y	**Telefon**	0,3962	0,4784	0,3884	0,5047	0,4634	0,3570
	MTw Awinda	1,1899	1,8843	2,0654	2,0573	1,1031	1,1725
Z	**Telefon**	2,0769	2,0014	1,9002	2,0014	1,7651	1,7949
	MTw Awinda	2,7939	2,4823	2,3164	2,8687	2,0401	2,5741
Różnica procentowa		**51,51%**	**59,94%**	**60,78%**	**52,67%**	**49,37%**	**50,20%**

Tabela 10. Wyniki badań maksymalnych bezwzględnych wartości przyśpieszeń liniowych, urządzenie Xiaomi Redmi Note 3 Pro

Oś	**System**	**MIUI**	**CyanogenMod**	**MoKee**	**Resurrection Remix OS**	**LineageOS**	**Flyme OS**
Xiaomi Redmi Note 3 Pro – wolna sekwencja ruchu [m/s2]							
X	**Telefon**	0,6908	0,4678	0,5281	0,5628	0,4073	0,4170
	MTw Awinda	1,4011	3,9149	2,3033	3,4341	1,1811	0,8708
Y	**Telefon**	0,4042	0,4259	0,3820	0,3994	0,4274	0,3434
	MTw Awinda	0,8419	4,0094	1,2284	1,9462	1,0472	0,8814
Z	**Telefon**	1,7938	1,7924	1,6027	1,6689	1,6866	1,7772
	MTw Awinda	1,4076	2,3930	1,3966	2,1711	1,5124	1,9377
Różnica procentowa		**41,41%**	**67,51%**	**52,94%**	**62,07%**	**45,01%**	**40,48%**

Tabela 11. Wyniki badań maksymalnych prędkości kątowych, urządzenie Xiaomi Redmi 3 Pro

Oś	**System**	**MIUI**	**CyanogenMod**	**MoKee**	**Resurrection Remix OS**	**LineageOS**	**Flyme OS**
Xiaomi Redmi 3 Pro – wolna sekwencja ruchu [rad/s]							
X	**Telefon**	2,2312	2,3199	2,3878	2,1283	2,2464	2,0367
	MTw Awinda	2,4985	2,4248	2,6996	2,9735	2,7031	2,3205
Y	**Telefon**	0,9332	1,3600	0,8617	0,7671	1,0608	1,0799
	MTw Awinda	1,2386	1,8748	1,4321	1,6106	1,3660	1,7297
Z	**Telefon**	1,8604	1,7769	1,8069	1,6969	1,4602	1,5768
	MTw Awinda	1,9631	1,8896	1,8054	1,9003	1,5257	1,5857
Różnica procentowa		**13,53%**	**12,58%**	**17,16%**	**30,50%**	**14,51%**	**16,78%**

Tabela 12. Wyniki badań maksymalnych prędkości kątowych, urządzenie Xiaomi Redmi Note 3 Pro

Oś	**System**	**MIUI**	**CyanogenMod**	**MoKee**	**Resurrection Remix OS**	**LineageOS**	**Flyme OS**
Xiaomi Redmi Note 3 Pro – wolna sekwencja ruchu [rad/s]							
X	**Telefon**	1,8111	1,7827	1,7857	2,3495	1,6916	1,7515
	MTw Awinda	2,2395	2,4174	2,7500	3,2029	2,1185	2,1834
Y	**Telefon**	0,5787	0,6570	0,6002	0,6383	0,8219	0,6211
	MTw Awinda	1,3406	1,3649	1,2963	1,4791	2,0043	1,2590
Z	**Telefon**	1,4679	1,6002	1,5177	1,5444	1,2763	1,2958
	MTw Awinda	1,6571	1,8793	1,7743	1,7334	1,4985	1,6303
Różnica procentowa		**29,13%**	**30,99%**	**23,95%**	**31,46%**	**31,32%**	**30,32%**

4. Wnioski

Analizę dokładności działania czujników wbudowanych wykonano dla sześciu różnych systemów operacyjnych. Z przeprowadzonych pomiarów i otrzymanych wyników można stwierdzić, że system ma wpływ na pracę sensorów. Dokładność pomiarów zależy od kilku czynników, między innymi jakości użytych komponentów, kalibracji, wykorzystanego algorytmu, typu ruchu. Badania pozwoliły również potwierdzić to, że czujniki pracują najlepiej pod kontrolą fabrycznego systemu telefonu.

Praca mierników światła urządzeń mobilnych uzależniona jest od panujących warunków oświetleniowych w otoczeniu. Używane są do kontroli podświetlenia wyświetlaczy LCD. Głównym przeznaczeniem czujników światła jest detekcja zmienności oświetlenia, a nie bezpośredni pomiar bezwzględnej wartości natężenia, co potwierdzają wyniki przeprowadzonych badań. Badania czujnika orientacji potwierdzają fakt, że czujnik pola magnetycznego Ziemi jest najbardziej podatny na zakłócenia. Wartości azymutu obarczone są dużymi błędami pomiarowymi. Kąty pochylenia i przechylenia charakteryzują się małym odchyleniem. Z przeprowadzonych pomiarów można wywnioskować, że żyroskop charakteryzuje się wysoką kulturą pracy w przeciwieństwie do akcelerometru. Pomimo dużej czułości na najmniejsze ruchy urządzenia, wykorzystywane są do sterowania aplikacjami, gdzie precyzja pomiarów nie odgrywa znaczącej roli.

Pomimo obserwowalnej najlepszej pracy czujników urządzeń mobilnych pod kontrola fabrycznego systemu MIUI, mają miejsce sytuację, w których zmodyfikowany system może stanowić pewną alternatywę do różnych zastosowań:

- CyanogenMod w pomiarze światła w warunkach naturalnego oświetlenia
- Flyme OS w pomiarze kąta przechylenia
- Lineage OS na urządzeniu Xiaomi Redmi 3 Pro do pomiaru wartości przyśpieszeń
- MoKee na urządzeniu Xiaomi Redmi Note 3 Pro do pomiaru wartości przyśpieszeń przy szybkich ruchach wykonywanych urządzeniem. Alternatywym systemem dla Xiaomi Redmi 3 Pro może być Lineage OS
- Flyme OS na urządzeniu Xiaomi Redmi 3 Pro do pomiaru prędkości kątowych przy szybkich ruchach wykonywanych urządzeniem

Badania te można rozszerzyć o dodatkowe dwa kierunki. Pierwszym z nich to ściślejsze określenie wpływu temperatury otoczenia na dokładność pomiarów czujników inercyjnych urządzenia mobilnego. Drugim kierunkiem mogłoby być badanie jak duży wpływ na jakość odczytów z sensorów pod kontrolą zmodyfikowanego systemu może mieć posiłkowanie się danymi pochodzącymi z innych czujników. Należałoby również wykonać dodatkowe badania w kierunku oceny jakości czujników wykorzystywanych w popularnych modelach telefonów. Duże zróżnicowanie w dokładności pomiarów może być spowodowane tym, że telefony Xiaomi zostały wyposażone w czujniki charakteryzujące się niską precyzją.

Literatura

[1] Global sales of smartphones in First Quarter 2016, http://www.gartner.com/newsroom/id/3323017[18.06.2017]

[2] Vassilis G., Perakis H., Rigorous Performance Evaluation of Smartphone GNSS/IMU Sensors for ITS Applications, Sensors (14248220). Aug2016, Vol. 16 Issue 8, p1-21. 21p.

[3] Smołka J., Skublewska-Paszkowska M., Badurowicz M., Łukasik E., Mobile devices as acceleration recorders, IAPGOŚ 1/2015, str. 49-52

[4] Mourcou Q., Fleury A., Franco C., Klopcic F., Vuillerme N. Performance Evaluation of Smartphone Inertial Sensors Measurement for Range of Motion, Sensors (14248220). Sep2015, Vol. 15 Issue 9, p23168-23187. 20p.

[5] Żyroskopy i akcelerometry MEMS w elektronice użytkowej, http://elektronikab2b.pl/technika/12098-zyroskopy-i-akcelerometry-mems-w-elektronice-uzytkowej#.WMf8aFU1_tQ [18.06.2017]

[6] John G. Webster, Eren H., Measurement, Instrumentation, and Sensors Handbook, CRC Press, 2014

[7] Poznajemy technologie: Czujniki w smartfonach i tabletach, https://lenovozone.pl/blog/poznajmy-technologie-czujniki-w-smartfonach-i-tabletach/ [18.06.2017]

[8] All about hall effect sensor, http://www.help2educate.com/hall-effect-sensor/ [18.06.2017]

[9] Sensors Overview- Android documentation, https://developer.android.com/guide/topics/sensors/sensors_overview.html [22.05.2017]

[10] GMS Arena – porównanie modeli telefonów, http://www.gsmarena.com/compare.php3?idPhone1=8007&idPhone2=7863 [18.06.2017]

[11] Instrukcja obsługi cyfrowego miernika światła EXTECH LT300, http://www.tme.eu/pl/Document/4802e0ccbdb6f91635a598ff91fe33f0/INSTR-LT300.pdf [18.06.2017]

[12] MTw User Manual, https://fccid.io/pdf.php?id=1949647 [18.06.2017]

[13] Graficzna prezenetacja układu współrzędnych, https://www.researchgate.net/figure/262055313_fig2_Average-roll-pitch-and-yaw-angles

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI X (2017) 200-203
Wysłane: 2017-06-29
Przyjęte: 2017-07-03

Analiza wpływu refactoringu na jakość kodu – analiza porównawcza dwóch przypadków

Mariusz Łukasik*, Marek Miłosz
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. w metodykach zwinnych jedną z technik poprawy jakości kodu jest jego refaktoryzacja. Jest to proces, wykorzystujący szereg technik, modyfikacji kodu bez zmiany jego funkcjonalności ukierunkowany na poprawę jego przejrzystości i zmniejszenie podatności na błędy. Poprawę kodu można mierzyć przy pomocy różnych metryk jakości kodu. w artykule przedstawiono analizę wpływu refaktoryzacji na statyczną jakość kodu na przykładzie open-sourcowego projektu Scuba. Jakość kodu została zmierzona w dwóch różnych punktach rozwoju oprogramowania – bezpośrednio przed i po refaktoryzacji kodu. Do pomiaru wykorzystano trzy najbardziej popularne zestawy metryk jakości kodu obiektowego i narzędzie Sonarqube. Badania wskazują jednoznacznie na istotną poprawę statycznej jakości kodu w wyniku refaktoringu.

Słowa kluczowe: refaktoryzacja; statyczna jakość kodu; metryki obiektowe

*Autor do korespondencji.

E-mail address: mrl1992@o2.pl

Analysis of the impact of refactoring on code quality – comparative analysis of two cases

Mariusz Łukasik*, Marek Miłosz
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. In agile methods, one of the techniques for improving code quality is refactoring. This is a process that employs a number of techniques, modifying the code without changing its functionality, aiming to improve its transparency and reduce vulnerability. You can measure the improvement of the code using different code quality metrics. The paper presents an analysis of the effect of refactoring on static code quality on the example of the open-source project Scuba. The quality of the code was measured at two different points of software development - right before and after refactoring the code. The three most popular sets of object code quality metrics and the Sonarqube tool were used for the measurement. The research clearly demonstrates the significant improvement of static code quality as a result of refactoring.

Keywords: refactoring, static code analysis, object metrics

*Corresponding author.

E-mail address: mrl1992@o2.pl

1. Wstęp

Zarządzanie projektami informatycznymi należy do skomplikowanych przedsięwzięć. Zawiera w sobie wiele działań i niesie ze sobą duże ryzyko niepowodzenia [1]. Osiągniecie sukcesu w obszarze IT wymaga konkretnej metodyki postępowania. Metodyki zwinne (*ang. Agile*) są jednym z przykładów metodyk wytwarzania oprogramowania, gdzie wykorzystywane są odmienne zasady i reguły postępowania, niż w powszechnie używanych metodykach tradycyjnych. Motywem stosowania metodyk zwinnych w zarządzaniu projektem jest precyzyjna i skuteczna reakcja na nowe problemy i wyzwania, pojawiające się w trakcie tworzenia oprogramowania. Dzięki elastycznemu podejściu, jakie oferują metodyki zwinne, łatwiejsze jest osiągnięcie wysokiego poziomu użyteczności biznesowej uzyskanej przez wytworzone i wdrożone oprogramowanie. Użytkownicy wytwarzanego oprogramowania w początkowym etapie prac projektowych często nie są w stanie precyzyjnie określić dokładnych wymagań i potrzeb związanych z oczekiwanym oprogramowaniem. w takiej sytuacji metodyki zwinne umożliwiają prowadzenie projektu i zakończenie go sukcesem, inaczej niż, gdyby użyto metodyk tradycyjnych, gdzie wszystkie wymagania odnośnie docelowego produktu muszą być skonkretyzowane już na początku prac projektowych [1]. Jedną z technik stosowanych w metodykach zwinnych wytwarzania oprogramowania jest refaktoryzacja kodu.

Refaktoryzacja (*ang. refactoring*) jest to proces wprowadzania zmian w strukturze projektu informatycznego nie mający wpływu na zmianę funkcjonalności ale poprawiający wydajność, koszty jego tworzenia, jakość oprogramowania [2]. Przez jakość oprogramowania rozumieć tu należy: testowalność, użyteczność, pielęgnację, zrozumienie, a także poziom osiąganej abstrakcji. Metod i zasad refaktoryzacji jest dużo i mogą być wyrażane w różnych poziomach abstrakcji. Przygotowywanie kodu na możliwe zmiany jest składnikiem idei programowania defensywnego. Jego podstawowym celem wydaje się być takie zabezpieczenie kodu, by zmiany lub nieprawidłowe wywołania w innej strefie projektu nie spowodowały szkód. Refaktoryzacja jest też możliwa na poszczególnych poziomach projektu takich jak: poziom danych, instrukcji, systemu , implementacji klasy lub interfejsu [2].

2. Metryki statycznej jakości kodu

Metryki statycznej jakości kodu są szybką i bardzo wygodną możliwością oceny jakości tworzonego oprogramowania. Znajomość ich wartości i wielkości pozwala na szacowanie złożoności, kosztu pielęgnacji oraz elastyczności systemu. Najbardziej popularne metryki obiektowe to: Martina, MOOD oraz Chidambera i Kemerera.

Metryki Chidambera i Kemerera powszechnie nazywane są zestawem metryk CK. Zostały one zaprojektowane w 1991 roku przez S.R. Chidambera i C.F. Kemerera [3]. Zawierają w sobie 6 metryk badających takie aspekty obiektowości, jakie do tej pory nie były używane w kontekście pomiarów oprogramowania: dziedziczenie, złożoność klasy, powiązania między klasami i ich spójność. Metryki te badają pojedyncze klasy i zbiory klas, stąd też służą jako narzędzie do niskopoziomowego badania jakości kodu obiektowego. Zestaw metryk CK składa się z następujących metryk [4]:

- **Depth of Inheritance Tree** (ang. *głębokość drzewa dziedziczenia, DIT*);
- **Response for a Class** *(*ang. *odpowiedź klasy, RFC;*
- **Lack of Cohesion Of Methods** (ang. *brak spójności metod,* LCOM);
- **Coupling Between Objects** (ang. *powiązania pomiędzy obiektami, CBO*);
- **Number of Children** (ang. *liczba klas pochodnych, NOC*);
- **Weighted Method per Class** (ang. *ważona liczba metod w klasie, WMC*).

Inaczej niż w przypadku metryk Chidambera i Kemerera, metryki Martina biorą pod uwagę pojęcie powiązania jednej klasy z drugą. z omówionymi tutaj metrykami wiążą się dwie zasady [5]. Są to zasada stabilne abstrakcyjne SAP (*ang. Stable Abstractions Principle*), a także stabilnych zależności SDP (*ang. Stable Dependency Principle*). w szeroko dostępnych narzędziach stosujących metryki używa się 5 następujących metryk [6]:

- **Afferent coupling (Ca)** – *(ang. powiązania do wewnątrz – dośrodkowe;*
- **Efferent coupling (Ce)** – *(ang. powiązania do zewnątrz - odśrodkowe);*
- **Abstractness (A)** – *(ang. abstrakcyjność);*
- **Instability (I)** – (*ang. niestabilność*);
- **Normalize distance from main sequence (Dn) -** (*ang. znormalizowana odległość od ciągu głównego*).

Metryki MOOD (*ang. Metrics of Object Oriented Design*) składają się z 6 metryk zaprezentowanych przez F. B. e Abreu w roku 1994 [7]. Kilka z nich zostało zmodyfikowanych i zaproponowanych w roku 1995 również przez tych samych autorów w takim kształcie jak obecnie są omawiane. Celem nadrzędnym tych metryk jest analiza stopnia użycia następujących paradygmatów obiektowych: hermetyzacji, powiązań, polimorfizmu, dziedziczenia. Przedmiotem pomiaru jest system jako całość, a nie konkretne klasy i relacje miedzy nimi lub pakietami. Metryki te dają zatem obraz jakości całego projektu, co więcej są mocno skalowalne i nie zależą od rozmiaru badanego systemu lub języka implementacji. Nie dają niestety obiektywnego wyniku w przypadku systemów, w których główną rolę grają formularze, generacja kodu i segmenty współdzielone przez cały system, w których podział odpowiedzialności do klas opiera się na różnych kryteriach [8]. Wszystkie metryki MOOD posiadają pewne cechy wspólne. Wyrażane są jako poziom wykorzystania w projekcie w stosunku do maksymalnego możliwego poziomu. z racji tego są to wartości wyrażane procentowo. Dzięki temu dają możliwość szybkiej oceny całości systemu, co jest przydatne głównie z przeznaczeniem dla poziomu zarządzania projektem.

Wśród metryk MOOD rozróżnia się:

- **Attribute Hiding Factor (AHF)** (ang. *współczynnik ukrycia atrybutów*);
- **Method Hiding Factor (MHF)** (ang. *współczynnik ukrycia metod*);
- **Attribute Inheritance Factor (AIF)** (ang. *współczynnik dziedziczenia atrybutów*);
- **Method Inheritance Factor (MIF)** (ang. *współczynnik dziedziczenia metod*);
- **Polymorphism Factor (PF)** (ang. *współczynnik polimorficzności*);
- **Coupling Factor (CF)** (ang. *współczynnik powiązań*).

3. Teza i metodyka badań

Refaktoryzacja ma między innymi służyć poprawie jakości kodu. Realnym zatem wydaję się postawienie następującej tezy: refaktoryzacja poprawia statyczną jakość kodu. w celu udowodnienia tej tezy wykonany został eksperyment. Polegał on na wyznaczeniu wartości metryk statycznej jakości kodu dla oprogramowania rozwijanego projektu w dwóch wersjach: bezpośrednio przed i po jego refaktoryzacji.

Obiektem badań był projekt Scuba. Kod pozyskany został z repozytorium projektu: https://github.com/xwangsd/Scuba. Scuba jest jedną z wersji frameworka SCUBA służącego do programowania kart inteligentnych (ang. S*mart Cards*). Autor dokonał na nim refactoryzacji polegającej na usunięciu części niepotrzebnego kodu, wydzieleniu kodu do osobnych klas itp. Wybór tego projektu uzasadniony jest ze względu na jego wielkość. Posiada on ponad 10 000 linii kodu a także ponad 200 klas.

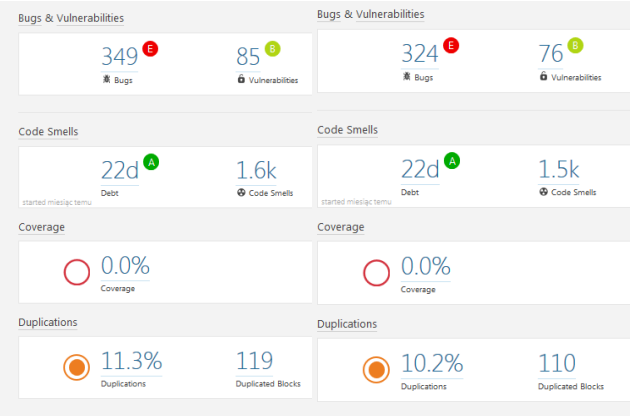
Do analizy jakości kodu użyte zostało narzędzie Sonarqube. Narzędzie to umożliwia kompleksową analizę kodu w ośmiu obszarach: niezawodność, bezpieczeństwo, łatwość utrzymania, pokrycie testami, duplikaty, parametry projektu, złożoność, problemy do rozwiązania. Wyniki w niektórych obszarach są oceniane literami od a do E, gdzie ocena (*ang. rating*) oznaczona literą a jest oceną najlepszą, a E – najgorszą.

4. Rezultaty badań

Celem badania będzie sprawdzenie tezy: ***Refaktoryzacja poprawia jakość kodu.***

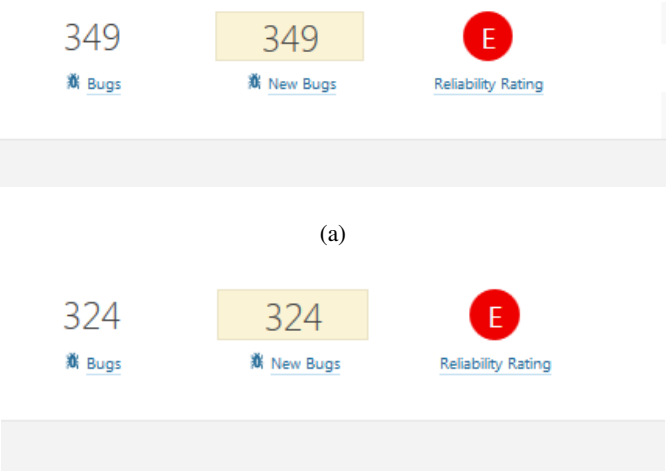
Przechodząc do szczegółowego omówienia analizy kodu podzielono je na 8 sekcji, pod kątem których badano projekt

Scuba: niezawodność, bezpieczeństwo, łatwość utrzymania, pokrycie testami, duplikaty, parametry projektu, złożoność, problemy do rozwiązania. Wyniki badań przedstawiają rys. 1-3.



(a) (b)

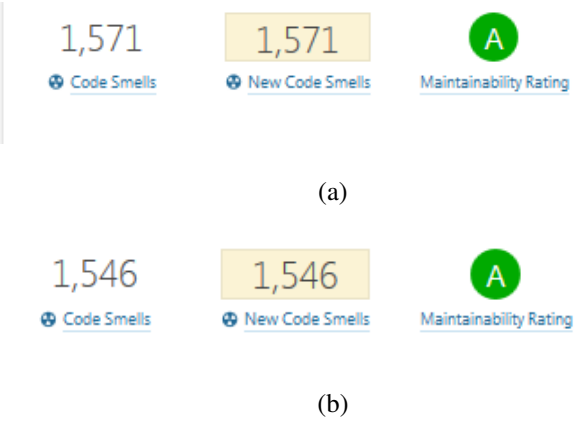
Rys. 1. Ogólna analiza jakości kodu projektu przed (a) i po (b) refaktoryzacji



(a)

(b)

Rys. 2. Widok dotyczący niezawodności projektu w panelu Sonarqube przed (a) i po (b) refaktoryzacji



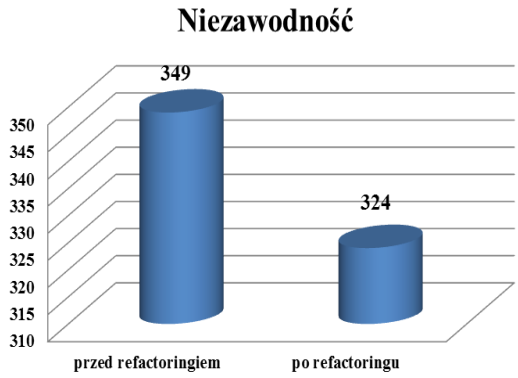
(a)

(b)

Rys. 3. Widok dotyczący łatwości utrzymania projektu w panelu Sonarqube przed (a) i po (b) refaktoryzacji

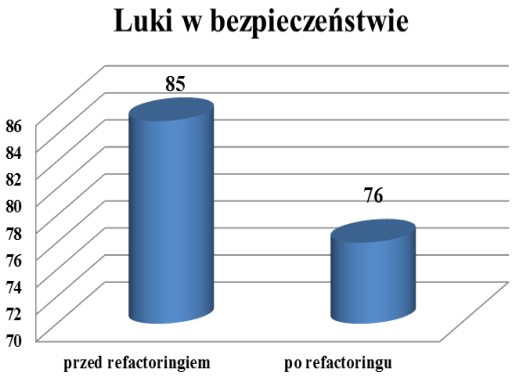
5. Analiza porównawcza

W niniejszym rozdziale dokonano analizy porównawczej wraz z dyskusją. Analiza porównawcza w formie graficznej podsumowuje badania. Na wykresach (rys. 4-8) przedstawiono zmiany w jakości kodu po dokonaniu refaktoryzacji.



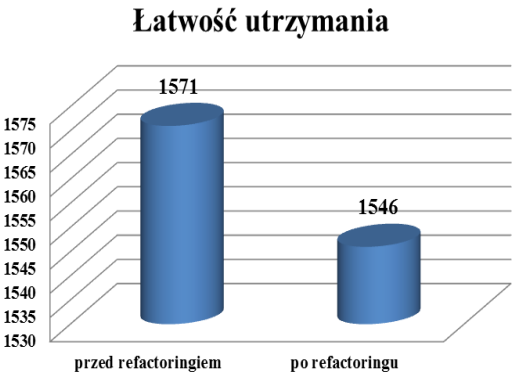
Rys. 4. Porównanie liczby błędów w kodzie

Niezawodność kodu mierzona liczbą błędów spadła po refaktoryzacji z 349 do 324 (rys. 4).



Rys. 5. Porównanie liczby luk w bezpieczeństwie

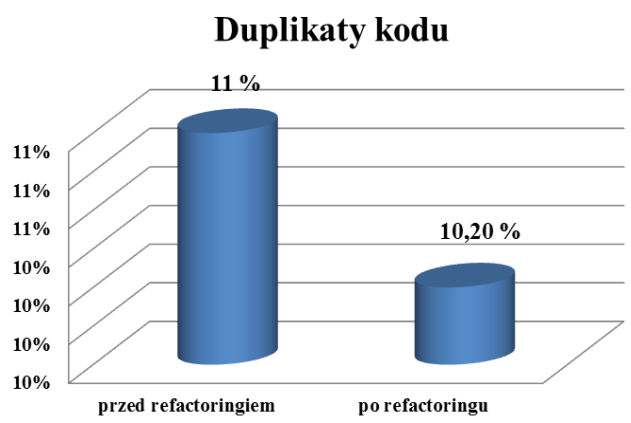
Liczba luk w bezpieczeństwie także spadła z 85 do 76 (rys.5).



Rys. 6. Porównanie łatwości utrzymania kodu

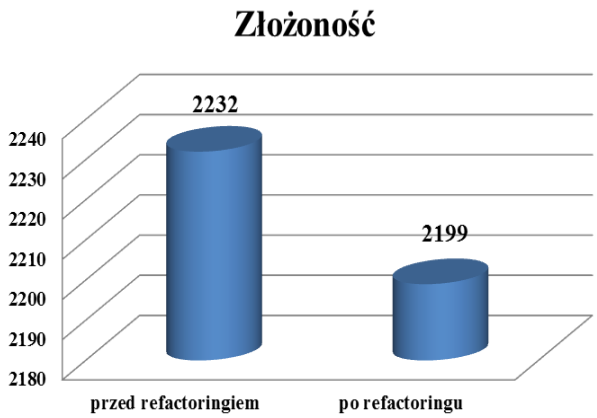
Analizując łatwość utrzymania projektu można dostrzec, że spadła ilość miejsc w kodzie wymagających zmiany

implementacji jako potencjalne miejsca przyszłych problemów z 1571 do 1546 (rys. 6).



Rys. 7. Porównanie ilości udziału duplikatów w kodzie

Stosunek kodu zduplikowanego do kodu unikalnego spadł z 11% do 10,2%. Jeśli chodzi o liczbę linii kodu to spadła z 1947 do 1795 (rys. 7).



Rys. 8. Porównanie złożoności kodu

Co ważne projekt stał się także mniej złożony. Ocena złożoności spadła z 2232 do 2199 pkt. (rys. 8).

6. Podsumowanie

W artykule dokonano przeglądu statycznych metryk analizy kodu. w części badawczej metryki zostały wykorzystane do badania projektu o nazwie Scuba pod kątem wpływu refaktoryzacji na jakość kodu. Omówiono również wyniki analizy a także przedstawiono wnioski.

Artykuł ten może być źródłem nauki metryk statycznej analizy kodu w kontekście praktycznym. Używanie metryk często wymaga doświadczenia i wyrobienia własnych doświadczeń. Metryki stanowią cenne wsparcie w procesie refaktoryzacji. Otwierają możliwość oceny jakości kodu na wielu poziomach abstrakcji i badania parametrów, które przekładają się na ogólną ocenę niezawodności i funkcjonalności systemu informatycznego.

Podsumowując warto skomentować postawioną wcześniej tezę, że refactoring wpływa na jakość kodu. Z analizy dokonanej za pomocą Sonarqube wynika jednoznacznie, że tak. Potwierdzają to m.in. pomiary złożoności projektu czy ilości duplikatów w kodzie ale także ocena niezawodności i łatwości utrzymania. Eksperyment badawczy wykorzystywał jeden projekt. By statystycznie udowodnić poprawność tezy należałoby zbadać większą liczbę projektów.

Literatura

[1] Mirosława Lasek, Aleksandra Adamus, *Informatyka ekonomiczna,* 2014.
[2] M. Miłosz, M. Borys, M. Plechawska-Wójcik, *Metodyki zwinne wytwarzania oprogramowania,* Politechnika Lubelska, 2011.
[3] S.R. Chidamber, C.F. Kemerer, a *metrics suite for object-oriented design,* IEEE Transactions on Software Engineering, No 6, Vol. 20, p. 476-493, 1994.
[4] Dr. Linda Rosenberg, *Applying and interpreting OO Metrics*, http://www.literateprogramming.com/ooapply.pdf.
[5] B. Henderson-Sellers, *Object-Oriented Metrics, measures of Complexity*, Prentice Hall, 1996.
[6] Robert Martin, *OO Design Quality metrics*. October 28, 1994.
[7] S.H. Kan, *Metryki i modele w inżynierii jakości oprogramowania,* PWN 2006, s. 355-383.
[8] F. B. Abreu, *The MOOD Metrics Set*. ECOOP Workshop on Metrics, 1995

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 204-207
Wysłane: 2017-06-29
Przyjęte: 2017-07-01

Porównanie efektywności szkieletów AngularJS i VueJS

Nazar Patrylo*, Marek Miłosz

Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule przedstawiono rezultaty analizy porównawczej szkieletów JavaScript AngularJS i VueJS dla tworzenia interfejsów użytkownika. Badanie wydajności zostało zrealizowane poprzez implementację, z użyciem obu szkieletów, aplikacji korzystających z języka JavaScript. Dla każdej z nich wykonano pomiary wydajności i na ich podstawie sporządzono analizę wyników. Analizę wykonano przy pomocy mechanizmu badawczego zawartego w opracowanym kodzie oraz przy użyciu narzędzi developerskich przeglądarek internetowych.

Słowa kluczowe: AngularJS; VueJS; JavaScripr szkielety; wydajność

*Autor do korespondencji.

Adres/adresy nazarpatrylo@gmail.com

Comparison of AngularJS and VueJS frameworks efficiency

Nazar Patrylo*, Marek Miłosz

Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article presents the results of a comparative analysis of JavaScript AngularJS and VueJS frameworks to create user interfaces. The study of productivity has been achieved through the implementation of JavaScript applications using both frameworks. Each of them was executed and the performance was measured. The results was a basis is the analysis. The analysis was done using the code build in the applications as well as the developer tools of web browsers.

Keywords: AngularJS; VueJS; JavaScript frameworks; efficiency

* Corresponding author.

1. Wstęp

Web-programowanie staje się coraz bardziej popularne ze względu na bardzo szybkie tempo rozwoju technologii informatycznych, w tym internetowych. Wraz z tym rozwojem, front-end programowanie przejmuje coraz więcej funkcji strony serwerowej, zmniejszając obciążenie serwera, a tym samym zwiększając ogólną wydajność aplikacji internetowych.

Najlepszym rozwiązaniem do automatyzacji i usprawnienia procesu tworzenia aplikacji stało się korzystanie przez programistów z różnych szkieletów JavaScript. Porządkuje to strukturę aplikacji oraz pozwala na ponowne wykorzystanie już istniejącego kodu. Pozwala to również zwiększyć skalowalność aplikacji [1].

Deweloperzy często wykorzystują wiele szkieletów i bibliotek do wykonywania swojej pracy, szczególnie przy opracowywaniu dużych i złożonych aplikacji. Jedną z ich głównych zalet jest to, że umożliwiają ponowne użycie kodu. Pozwala to na poświęcenie więcej czasu i uwagi na projektowanie i testowanie aplikacji internetowych. Z drugiej strony, istnieje również niebezpieczeństwo, że wybór nieodpowiedniego szkieletu spowoduje obniżenie jakości aplikacji oraz wydłuża terminy jej realizacji.

W ostatnim czasie JavaScript został nazwany jednym z najlepszych języków programowania, jakich można uczyć się w 2017 roku [2].

Popularność języka JavaScript szybko rośnie. W 2016 roku nastąpiły w nim wielkie zmiany. Całkowicie zmodernizowano i wdrożono drugą wersję angularjs, rozwinięto biblioteki jQuery, której używa aż 96,5% wszystkich stron internetowych wykorzystujących JavaScript, zmodernizowano standard ecmascript oraz zaktualizowano Node.js [3].

W tym artykule przedstawiony został proces analizy dwóch szkieletów AngularJS, VueJS. Kryterium wyboru tych frameworków jest ich ogromna popularność oraz sukces prestiżowych aplikacji napisanych za ich pomocą.

2. Cel i hipotezy pracy

Celem pracy jest analiza i porównanie szkieletów AngularJS i VueJS w procesie tworzenia interfejsów użytkownika za pomocą języka programowania JavaScript.

W celu zrealizowania założeń pracy sformułowano dwie hipotezy:

Hipoteza 1: Szkielet AngularJS 2 pokazuje lepsze wyniki przetwarzania danych aplikacji internetowych niż framework VueJS.

Hipoteza 2: Tworzenie aplikacji w szkielecie AngularJS 2 jest bardziej pracochłonne od tworzenia z wykorzystaniem szkieletu VueJS.

By udowodnić lub odrzucić hipotezy badawcze powstała specjalna aplikacja testowa. Przy jej pomocy przetestowano szybkości wykonania różnych operacji. W celu weryfikacji drugiej hipotezy poddano analizie utworzony kod (wraz ze szkieletem) i wyznaczono jego wielkość, mierzoną liczbą linii kodu.

3. Metodyka badań

3.1. Analiza metryk kodu

Zastosowanie testów metryk kodu pozwala kierownikom projektów i deweloperom oceniać różne aspekty tworzonego lub istniejącego produktu, przewidzieć wielkość wykonanej pracy, charakteryzować złożoność oprogramowania i wydajność zespołów je tworzących, oszacować ilość czasu i liczbę osób niezbędnych do napisania programu [8].

Najprostszą metryką jest liczba wierszy w kodzie (ang. *SLOC, Source Lines Of Code*) [7]. Użycie tej metryki jest utrudnione, ze względu na fakt, że jedna i ta sama funkcjonalność może być zapisana w jednym wierszu lub podzielona na kilka. Spowodowało to konieczność rozróżnienia pojęć fizyczna i logiczna linia kodu. Liczba linii kodu oprogramowania zależy także od stylu programowania i używanego języka programowania [4].

SLOC jako miara wielkości oprogramowania może być z powodzeniem użyta do porównania dwóch programów tworzonych przez tego samego programistę.

3.2. Analiza efektywności

Efektywność pracy, jest najważniejszym parametrem programu komputerowego. Dla porównania wydajności aplikacji internetowej zrealizowanej w języku JavaScript najlepiej skorzystać z funkcjonalności samej przeglądarki, ponieważ przeglądarka jest interpreterem kodu aplikacji webowej i ma dostęp bezpośrednio do charakterystyk wydajności jej wykonania.

Przeglądarka Google Chrome posiada zaawansowane narzędzie do sprawdzania wydajności strony – Dev tool. To narzędzie jest używane, m.in. do szczegółowej analizy czasu ładowania elementów na stronie. Program zawiera wszystkie niezbędne opcje do wglądu w pobieranie zasobów i wykonywanie kodu języka JavaScript.

Można stwierdzić, że wyżej opisane narzędzie daje duże możliwości testowania, debugowania, a także analizy poszczególnych elementów na stronie. W procesie realizacji eksperymentu skorzystano z dwóch narzędzi:

- Console, pozwalającego na przeglądanie różnego rodzaju ostrzeżeń i zaleceń, a także błędów.
- Performance, umożliwiającego przegląd czasu poświęconego na ładowanie strony.

3.3. Kryteria i metody porównania

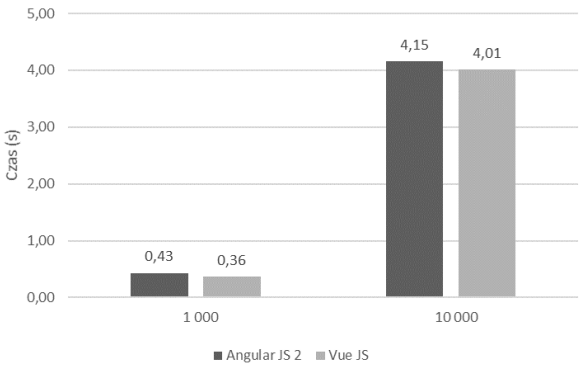
W celu porównania dwóch szkieletów zaimplementowano kilka typowych operacji aplikacji internetowej oraz przeprowadzono testy ich wydajności. Do typowych operacji zaliczone zostały: tworzenie, aktualizowanie, usuwanie obiektów na stronie i mierzenie czasu reakcji przy kliknięciu w obiekt. Badania przeprowadzono czterokrotnie w dwóch etapach: na małej oraz na dużej próbce danych. Z zebranych wartości wyliczono wartości średnie wykonania każdej z wyżej wymienionych operacji.

Drugi etap, czyli porównanie metryki kodu pozwala programiście określić przybliżony czas wykonania zadania, jego stopień trudności oraz scharakteryzować zastosowane wzorce projektowe.

4. Wyniki testów wydajności

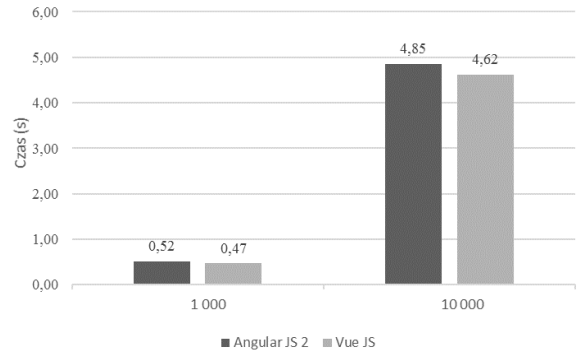
W testach zestawiono dwa programy napisane w różnych szkieletach, co pozwala porównać ich skuteczność i przeanalizować zalety i wady frameworków .

Po przeprowadzeniu eksperymentu i uzyskanych wyników można przystąpić do statystycznej analizy. Jej rezultaty dla obydwóch wersji aplikacji przedstawiono na rys. 1-6.



Rys. 1. Średnie czasowe charakterystyki tworzenia obiektów ekranowych dla różnych ich ilości

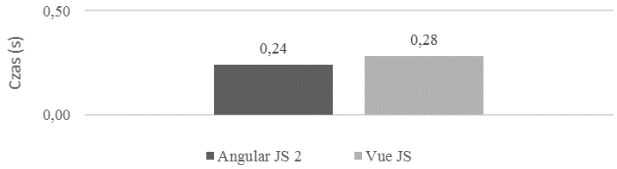
W teście przeprowadzonym na małych liczbach obiektów (informacji) różnica w wydajności szkieletów jest bardzo niewielka, ale przy teście na dużej liczbie różnica ta jest już zauważalna. Znacznie lepiej radzi sobie w tym zakresie szkielet VueJS Dotyczy to zarówno procesu tworzenia nowych obiektów na stronie (rys. 1), jak i dodawania dużej ilości informacji na stronie (rys. 2).



Rys. 2. Średnie czasowe charakterystyki dodawania obiektów ekranowych dla różnych ich wartości

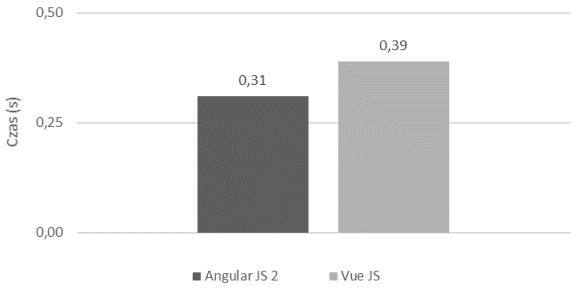
Drugim testowanym procesem jest aktualizacja elementów na stronie oraz testowanie aktualizacji każdego dziesiątego elementu w aplikacji (rys. 3), a także eksperyment zamiany kolejności obiektów w aplikacji (rys. 4).

Jak widać z rysunku 3, dla procesu aktualizacji każdego dziesiątego elementu, aplikacja napisana z wykorzystaniem szkieletu AngularJS 2 wykazywała lepsze rezultaty.



Rys. 3. Średnie czasy aktualizacji elementów interfejsu

W eksperymencie zmian sekwencji elementów aplikacja napisana we frameworku AngularJS 2, podobnie jak w poprzednim teście, osiągnęła lepszy wynik niż aplikacja oparta na szkielecie VueJS (rys. 4).

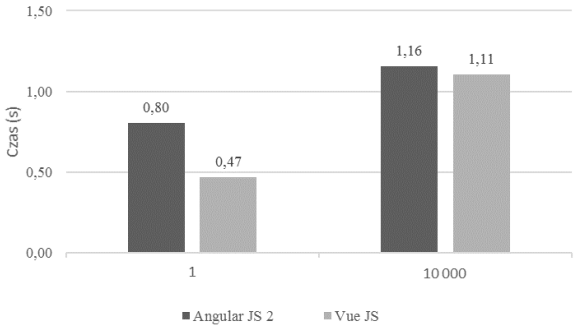


Rys. 4. Średnie czasy zmiany pozycjonowania elementów interfejsu

Trzecim testowanym elementem jest szybkość usuwania obiektów ze strony. Ten eksperyment był podzielony na dwie części.

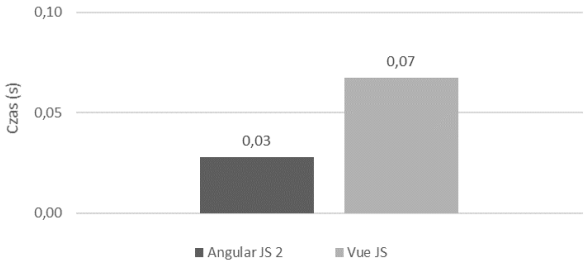
W pierwszej części zbadano szybkość usuwania pojedynczych elementów. Jak widać z rysunku 5 lepsze wyniki osiągnęła aplikacja napisana we frameworku AngularJS 2.

W drugiej części przeprowadzono analogiczny test, jednak liczba usuniętych elementów na stronie wynosiła 10000 sztuk. W tym przypadku lepsze wyniki osiągnęła aplikacja napisana w szkielecie VueJS.



Rys. 5. Średnie czasy usuwania obiektów

Czwartym testowanym elementem był czas reakcji na zdarzenie kliknięcia w elementy aplikacji. Eksperyment został zrealizowany przy zastosowaniu dużej liczby elementów na stronie. Jak widać z rysunku 6 aplikacja napisana na szkielecie AngularJS 2 osiągnęła ponad dwa razy lepszy wynik wydajności niż aplikacja oparta na szkielecie VueJS.



Rys. 6. Średnie czasy reakcji na kliknięcie na obiekcie ekranowym

Framework AngularJS 2 okazał się lepszy w przypadku: aktualizacji danych i czasie reakcji po naciśnięciu w elementy aplikacji, a framework VueJS w sytuacji tworzenia obiektów na stronie i usuwania elementów.

5. Wyniki eksperymentu analizy metryki kodu

Eksperyment badania liczby linii kodu obydwu aplikacji został zrealizowany za pomocą NodeJS wtyczki cloc - pozwalającej przeskanować projekt i przedstawić wyniki metryki kodu. W tabele 1 przedstawione zostały wyniki dla obu frameworków.

Tabela 1. LOC aplikacji testowych

Typ pliku	**VueJS**	**Angular JS 2**
JavaScript	140	416
TypeScript	-	206
Vuejs Component	137	-
JSON	32	61
HTML	12	55
Razem	**321**	**738**

Z wyników przeprowadzonego testu (tab. 1 i tab. 2) można wyciągnąć wnioski, że aplikacje napisane z wykorzystaniem szkieletu AngularJS 2 są bardziej pracochłonne niż aplikacje zrealizowane za pomocą szkieletu VueJS.

6. Wnioski

Na podstawie przeprowadzonych badań zostały wyciągnięte wnioski dotyczące efektywności pracy i wytwarzania aplikacji wykorzystujących szkielety AngularJS 2 i VueJS.

Pierwszym istotnym parametrem, na który trzeba zwrócić uwagę to jest analiza wartości metryki kodu (w tym przypadku LOC). Z jej wyników można wykryć, że aplikacje napisane na szkielecie VueJS są mniej pracochłonne niż aplikacje zrealizowane przy pomocy szkieletu AngularJS 2.

Na podstawie przeprowadzonych testów wydajności można zaobserwować, że szkielet AngularJS 2 lepiej radzi sobie w przypadku aktualizacji danych i reakcji na zdarzenie naciśnięcia w elementy programu. Z kolei VueJS poradził sobie znacznie lepiej w przypadku tworzenia nowych elementów na stronie i ich usuwania, co jest kryterium dużo istotniejszym, niż lekkość samego szkieletu.

Początkowo zakładano, że szkielet AngularJS 2 osiąga lepsze wyniki w przypadku przetwarzania danych w aplikacji internetowej niż VueJS (Hipoteza 1) i że aplikacje napisane na szkielecie AngularJS 2 są bardziej czasochłonne niż aplikacje zrealizowane za pomocą szkieletu VueJS (Hipoteza 2). Na podstawie przeprowadzonych badań otrzymano wyniki dementujące prawdziwość hipotezy pierwszej i potwierdzające słuszność hipotezy drugiej.

Literatura

[1] Mariano C. L.: Benchmarking JavaScript Frameworks. Masters dissertation, 2017. doi:10.21427/D72890

[2] Cristine Felt: Which Are The Best Programming Languages to Learn in 2017 , https://www.ibm.com/developerworks/community/blogs/d13a8e32-0870-49cb-9b0dba0e34fa6561/entry/Which_Are_The_Best_Programming_Languages_to_Learn_in_2017?lang=en

[3] Noon H.: JavaScript Frameworks in 2017, https://hackernoon.com/5-best-javascript-frameworks-in-2017-7a63b3870282

[4] Программный код и его метрики: https://habrahabr.ru/company/intel/blog/106082 [28.04.2017]

[5] Метрики сложности кода: http://www.ispras.ru/preprints/docs/prep_25_2013.pdf [01.05.2017]

[6] Source lines of code: https://en.wikipedia.org/wiki/Source_lines_of_code [07.06.2017]

[7] Oviedo E.J.: Control flow, data flow and program complexity, Chicago, 1980.

[8] Rentrop J.: Software Metrics as Benchmarks for Source Code Quality of Software Systems, Amsterdam, 2006

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 208-212
Wysłane: 2017-06-21
Przyjęte: 2017-06-23

Porównanie wydajności platformy Xamarin oraz aplikacji natywnych na przykładzie systemu operacyjnego Android

Ihor Bodia*, Małgorzata Plechawska-Wójcik

[a] Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule przedstawiono rezultaty analizy porównawczej poszczególnych narzędzi do wytwarzania aplikacji dla systemu operacyjnego Android. Analizę wydajnościową przeprowadzono w oparciu o aplikacje wytworzone przy pomocy poszczególnych frameworków, w szczególności Xamarina oraz natywnych narzędzi wytwarzania oprogramowania. Zdefiniowano kilka kryteriów badawczych, w oparciu o które otrzymano wyniki porównania wydajności i sformułowano wnioski.

Słowa kluczowe: Android; wydajność; porównanie; Xamarin; Java

*Autor do korespondencji.

Adres e-mail: ihor.bodia@pollub.edu.pl

Performance comparison of the Xamarin platform and native applications for Android operating system

Ihor Bodia*, Małgorzata Plechawska-Wójcik

[a] Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. The article presents the results of comparative analysis of various tools for creating applications for the Android operating system. Performance analysis was based on applications created using individual frameworks, in particular Xamarin and native software development tools. Several research criteria have been defined, based on which results of performance comparison have been derived and formulated conclusions.

Keywords: Android; performance; comparison; Xamarin; Java

*Corresponding author.

E-mail address: ihor.bodia@pollub.edu.pl

1. Wstęp

Praca ta zawiera teoretyczne oraz praktyczne informacje o dość nowym frameworku rozwoju oprogramowania jakim jest Xamarin, którego celem jest ujednolicenie rozwoju oprogramowania. W dzisiejszych czasach technologie informatyczne rozwijają się w szybkim tempie i wciąż pojawiają się nowe rozwiązania, począwszy od nowego języka programowania, kończąc nowymi rozwiązaniami sprzętowymi. Aby deweloperzy korzystający z platformy .NET mogli maksymalnie wykorzystać swoje możliwości w zakresie tworzenia oprogramowania na rynku wieloplatformowych rozwiązań został przedstawiony framework Xamarin. Ten framework jest doskonałym przykładem na to, jak firmy w branży rozwoju oprogramowania mobilnego konkurują między sobą, poprzez zapewnienie twórcom oprogramowania znanych interfejsów dla nieznanych urządzeń fizycznych lub platform mobilnych.

Celem pracy jest realizacja badania polegającego na porównywaniu wydajności działania platformy Xamarin i natywnych technologii budowy aplikacji dla systemu Android. Porównanie zrealizowane będzie w oparciu o wybrane kryteria analizy i wykonane na przykładzie dwóch funkcjonalnie identycznych aplikacji napisanych za pomocą wieloplatformowego frameworka Xamarin i natywnych narzędzi tworzenia aplikacji mobilnych dla systemu operacyjnego Android

W dzisiejszych czasach technologie informatyczne rozwijają się w szybkim tempie i wciąż pojawiają się nowe rozwiązania, począwszy od nowego języka programowania, kończąc nowymi rozwiązaniami sprzętowymi. Aby deweloperzy korzystający z platformy .NET mogli maksymalnie wykorzystać swoje możliwości w zakresie tworzenia oprogramowania na rynku wieloplatformowych rozwiązań został przedstawiony framework Xamarin[1].

Ten framework jest doskonałym przykładem na to, jak firmy w branży rozwoju oprogramowania mobilnego konkurują między sobą, poprzez zapewnienie twórcom oprogramowania znanych interfejsów dla nieznanych urządzeń fizycznych lub platform mobilnych.

Każda platforma ma własną charakterystykę i w związku z tym bardzo trudno jest stworzyć naprawdę uniwersalny interfejs, który pozwalał by w 100% wykorzystać możliwości danej platformy. Wspólnym wyznacznikiem dla takich interfejsów jest wskaźnik przenośności kodu, który określa jaka ilość kodu jest znormalizowana dla wszystkich środowisk i nie wymaga zmian podczas tworzenia aplikacji dla innych platform [2]. Każdy producent takich środowisk programowania wieloplatformowego stara się przybliżyć ten wskaźnik do 100%. Jednym z takich frameworków jest Xamarin, który został powołany w celu ujednolicenia wszystkich platform w jeden interfejs. Jak zostało wspomniane, trudno stworzyć naprawdę uniwersalny interfejs

[3]. Xamarin też ma swoje ograniczenia, przy wytwarzaniu oprogramowania, ponieważ wirtualna maszyna Java nie pozwała korzystać z dynamicznych obiektów takich jak *LINQ* [5].

2. Metodyki porównania środowisk

W aplikacjach które opracowano do testowania wydajności, dodano kilka testów symulujących niektóre obciążenia systemu. W oparciu o nie można uzyskać wyniki dotyczące wydajności i wyciągnąć pewne wnioski. Aplikacje będą posiadały te same zestawy funkcjonalności. Zostaną utworzone one za pomocą języków Java oraz C# w odpowiednich środowiskach Android Studio i Visual Studio dla systemu operacyjnego Android. Pomiar czasu testów aplikacji odbywa się za pomocą natywnych dla używanej platformy klas i metod pomiaru czasu.

Analiza ilości kodu. Analiza porównania kodu wykonana została za pomocą programu „*LocMetrics*" [4]. Ten program pozwala przeprowadzić analizę kodu według różnych kryteriów.

Szybkość generowania graficznego interfejsu użytkownika. Ten test pokazuje prędkość generowania niestandardowych elementów sterowania tworzonych dynamicznie w trakcie działania aplikacji. Podstawowa idea tego testu polega na dodaniu do kontenera *ListView*, elementu podrzędnego *ListViewItem*. Liczba dodawanych elementów jest zdefiniowana przez użytkownika w głównym widoku tego testu. Liczba elementów służąca do generowania interfejsu użytkownika wprowadzona została w liczbie stu tysięcy.

Przetwarzanie złożonych zapytań do bazy danych. Ten test pokazuje prędkość działania i wykonywania operacji *CRUD* (*ang. Create, Update Delete*) w aplikacji. System operacyjny Android pozwala korzystać z lekką bazę danych *SQLite*, która zostanie wykorzystana w tym teście. Dla ułatwienia pracy z tą bazą danych i podążać koncepcji OOP(*ang. Object-oriented programming*) podczas pracy z nią, zostaną wykorzystane ORM-narzędzie *ORMLite*, które są dostępne dla obu platform. Osłona *ORMLite* oferuje obiektowy interfejs pracy z bazą danych *SQLite*. Podczas korzystania z tego narzędzia, należy określić najpierw model tabeli bazy danych, która zostanie wykorzystana w teście.

Analiza pracy pętli z dużą ilością iteracji. Test ten jest dość prosty, sprawdzany jest czasu trwania cyklu z dużą liczbą powtórzeń. Liczba iteracji jest określana przez użytkownika na widoku tego testu, wprowadzając liczby. Głównym celem tego badania jest potwierdzenie lub obalenie hipotezy o tym, że Xamarin nie ustępuje w działaniu aplikacja na telefony narzędzi rozwoju, mimo że Xamarin jest dodatkowym poziomem abstrakcji nad środowiskiem natywnym. Podczas wykonywania tego testu, podjęto decyzję o tym jak testować pętle w warunkach wykonywania stu milionów iteracji.

Porównanie pracy kamery. W tym teście, prowadzone są badania w pracy kamery na różnych platformach. Głównym wskaźnikiem, na którym odbywa się porównania jest liczba klatek na sekundę, które są generowane przez aparat. Główna idea testu polega na wywołaniu kamery z głównego widoku testu, po czym, użytkownik jest zmuszony nagrać film. Po nagraniu wideo, plik filmu jest poddawany przetwarzaniu przez klasą *MediaExtractor*. Klasa ta pozwala wyodrębnić z pliku wideo metadane, wśród których dostępna jest wartość liczby klatek na sekundę. W trakcie badań nagrano trzy pliki wideo za pomocą wywołania kamery z natywnej aplikacji i trzy wideo za pomocą wywołania kamery z aplikacji opartej na frameworku Xamarin.

Analiza możliwości przetwarzania plików. W tym teście sprawdzane są możliwości pracy z plikami i wykonanie pomiarów ich konwersji. Zasada testu opiera się na konwersji plików binarnych do formatu ZIP i odwrotnie. Podczas działania tego testu, system odczytuje binarną zawartość plików, kompresuję i zapisuje pliki w lokalnym repozytorium. Do wykonania tego testu, z góry zostało wygenerowane tysiąc plików binarnych w różnych rozmiarach od zera do tysięcy bajtów. W tym teście, jest pomiar czasu kompresji plików oraz czas dekompresji.

3. Obiekt badań

Obiektami badań w pracy zostały dwie aplikacje, które były wytwarzane za pomocą frameworku Xamarin oraz natywnych narzędzi wytwarzania oprogramowania. W aplikacjach które zostały opracowane do testowania wydajności, dodane zostały kilka testów, symulujących niektóre obciążenia systemu, w wyniku działania których można uzyskać wyniki dotyczące wydajności i wyciągnąć pewne wnioski. Aplikacje będą posiadały te same zestawy funkcjonalności. Zostaną utworzone one za pomocą języków Java oraz C# w odpowiednich środowiskach Android Studio i Visual Studio dla systemu operacyjnego Android. Pomiar czasu testów aplikacji odbywa się za pomocą natywnych dla używanej platformy klas i metod pomiaru czasu.

4. Metoda badawcza

Porównanie aplikacji testowych zrealizowane będzie w oparciu o wybrane kryteria analizy i wykonane na przykładzie dwóch funkcjonalnie identycznych aplikacji napisanych za pomocą wieloplatformowego frameworka Xamarin i natywnych narzędzi tworzenia aplikacji mobilnych dla systemu operacyjnego Android.

W pracy postawiono następującą tezę: Platforma Xamarin zapewnia bardziej wydajny sposób budowy aplikacji dla systemu operacyjnego Android niż oprogramowanie natywne.

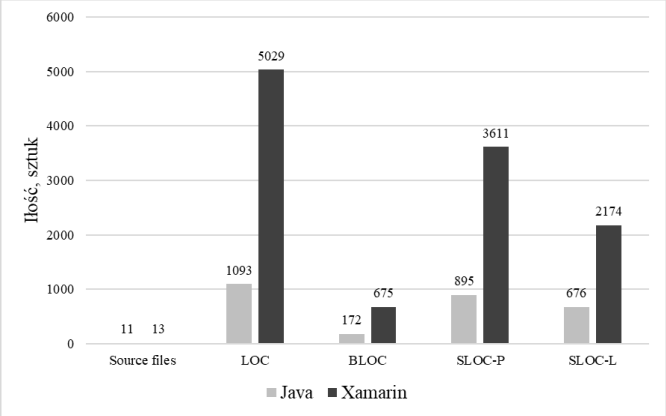
Hipoteza 1: Framework Xamarin generuje więcej kodu dla tego, że nie jest natywnym narzędziem do tworzenia aplikacji do systemu Android.

Hipoteza 2: Xamarin teoretycznie może być wolniejszy w kontekście szybkości generowania graficznego interfejsu użytkownika i przetwarzania multimediów, dla tego, że wywołuje bezpośrednio funkcje Javy.

Aby analiza była bardziej oczywista i jawna, do oceny wyników badań będą wykorzystywane liczby zwane wagami, Każda waga określa częstotliwość wykorzystania danego testu w stosunku do innych testów. Głównym celem danego testu jest ustalenie sumarycznej liczby wag dla każdej platformy. Zakres liczb będzie od 1 – najmniej stosowane do 3 – najwięcej stosowane.

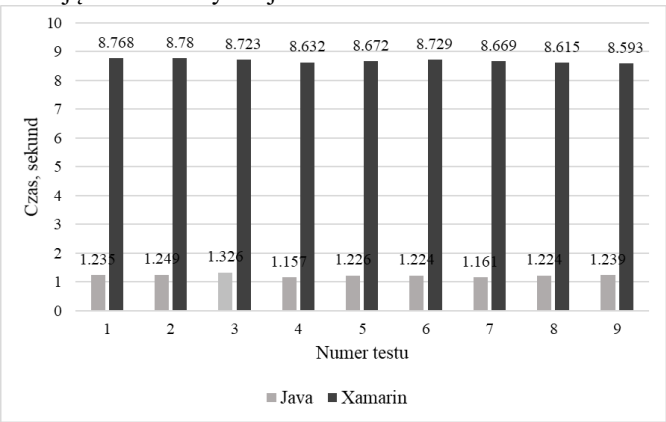
5. Rezultaty badań

Analiza rezultatu pomiarów ilości kodu. Analiza ilości kodu pokazała ogromna różnicę między dwoma projektami. Aplikacja napisana przy pomocy natywnych narzędzi, zajmuje mniej linii kodu oraz ma bardziej zoptymalizowane pliki z punktu widzenia ilości kodu, niż aplikacja napisana w języku C#. Przede wszystkim wynika to z cech składni języków C# i Java. Należy zauważyć, że tworzenie aplikacji wieloplatformowych za pomocą Xamarin, wymaga dodania do zwykłej natywnej aplikacji dodatkowej warstwy abstrakcji, co odbija się na wydajności i liczbie dodatkowo wygenerowanych plików.



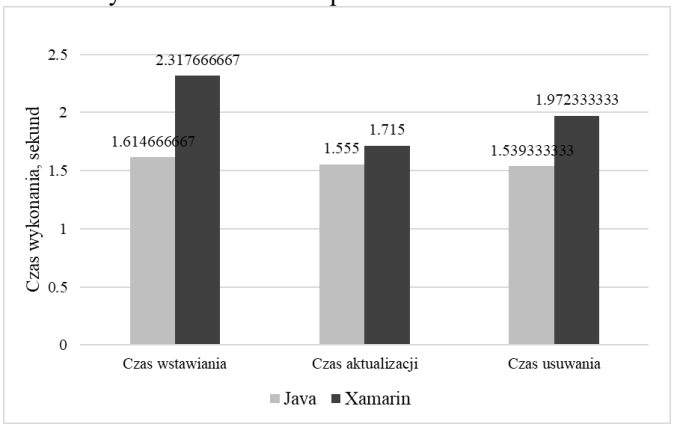
Rys. 1. Porównanie metryk kodu

Analiza rezultatów dynamicznego generowania interfejsu użytkownika. Podczas przeprowadzania testu tworzenia graficznego interfejsu użytkownika, okazało się, że dynamiczne generowanie interfejsu użytkownika szybciej odbywa się w natywnych środowiskach wykonawczych z systemem Android. Można przypuszczać, że dzieje się tak dlatego, że natywne środowisko wymaga na etapie kompilacji informacji na temat kontrolowanych obiektów. Różnica okazała kolosalna, aplikacja na podstawie frameworka Xamarin generuje elementy graficzne użytkownika sześć-siedem razy wolniej niż środowisko natywne. W wyniku badań, należy zauważyć, że mechanizmy natywnych narzędzi działają znacznie szybciej.



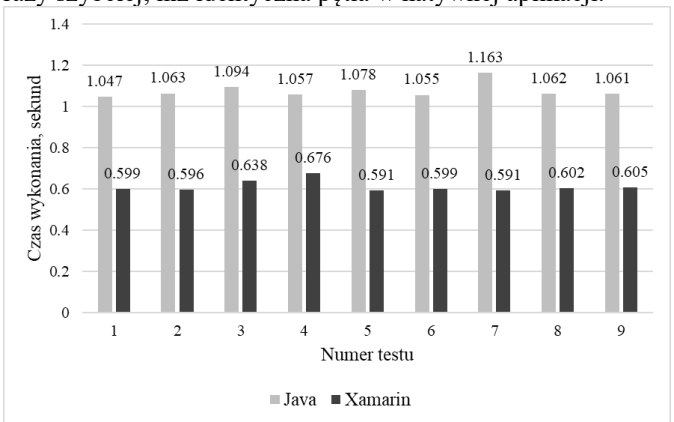
Rys. 1. Porównanie wyników testowania generowania graficznego interfejsu użytkownika

Porównanie rezultatów działania bazy danych. W teście z badaniem wydajności wbudowanych baz danych została sprawdzona szybkość wykonywania zapytań do bazy danych. Warunki testu były takie same w obu aplikacjach - wykorzystano obiektowe narzędzie *SQLite* do bezpiecznego korzystania z narzędzi bazy danych. W wyniku przeprowadzonych testów, okazało się, że natywne środowisko działa szybciej, choć różnica nie jest duża. Wyniki testów nie były nieoczekiwane, jak wspomniano wcześniej, nad natywnym środowiskiem wykonania jest środowisko wykonania Mono [3], która wymaga dodatkowych nakładów czasu procesora.



Rys. 2. Porównanie wyników testowania *CRUD* operacji

Analiza wyników testu pętli z dużą ilością iteracji. Analiza działania pętli z dużą liczbą iteracji przedstawiła niejednoznaczne wyniki. Działanie pustej pętli w aplikacji na bazie frameworka Xamarin okazało się szybsze, niż w zwykłej aplikacji napisanej przy pomocy języka Java, pomimo tego, że liczba iteracji była sto milionów. Biorąc pod uwagę to zachowanie, można przypuszczać, że wynikało ono z działania kompilatora .NET który optymalizuje kod na etapie kompilacji w wyniku czego puste wywołanie cyklu zostało zoptymalizowane. Ostatecznie, wyniki były takie, że pętla w aplikacji na bazie frameworka Xamarin działa o dwa razy szybciej, niż identyczna pętla w natywnej aplikacji.

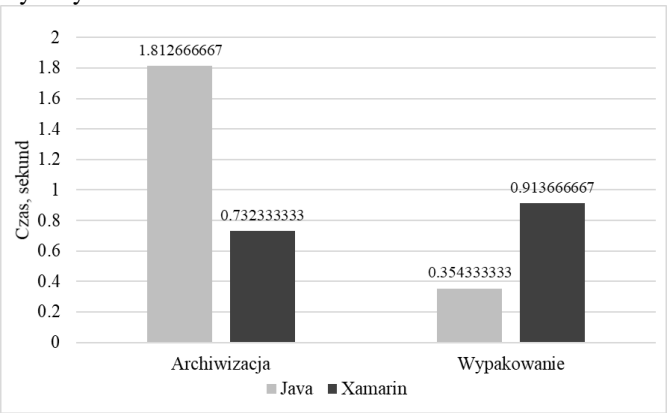


Rys 3. Porównane wyniki testowania czasu działania pustej pętli

Analiza wyników pracy kamer. Podczas porównywania pracy kamer została stworzona przez dwa dość proste skrypty do testowania. Aby rozpocząć test, użytkownik musiał nagrać film, a później system analizował go. Kryterium, według którego była porównana wydajność zostało oparte o liczbę klatek na sekundę każdego nagranego wideo. W efekcie końcowym okazało się, że używana platforma nie ma wpływu na pracę kamery. Należy przypuszczać, że wynika to z faktu, że w tym teście zostały wykorzystane mechanizmy intencji, co oznacza, że obiekt

kamery jest częścią serwisu, który dostarcza informacji o kamerze zainstalowanej na danym urządzeniu. Na tej podstawie można stwierdzić, że środowisko wykonania, z którego została wywołana kamera w żaden sposób nie wpływa na jej wydajność, tak jak ma to miejsce w przypadku wywołania obiektu kamery, która jest wykorzystywana przez zewnętrzną aplikację systemową.

Badanie rezultatów testu przetwarzania plików. Podczas badania możliwości środowisk uruchomieniowych oraz pracy z przetwarzaniem plików, został porównany czas konwersji plików binarnych do formatu ZIP i odwrotnie. Przede wszystkim, warto zauważyć, że natywne narzędzie programistyczne nie zawiera interfejsu, który oferowałby wszystkie niezbędne funkcje do pracy z kompresją plików w formacie ZIP. Wyniki porównania są niejednoznaczne. Czasy archiwizacji okazały się dwa razy lepsze niż aplikacja bazująca na frameworku Xamarin, podczas gdy czas rozpakowania okazał się lepszy w natywnych narzędziach wykonywania.



Rys. 4. Porównanie wyników czasu przetwarzania plików

Po przeprowadzeniu wszystkich testów, porównań i analiz należy zauważyć, że wyniki okazały się mieszane. Przede wszystkim warto zauważyć, że obie hipotezy były potwierdzone, a mianowicie, w przypadku tworzenia aplikacji dla systemu operacyjnego Android za pomocą Xamarin, generowane jest znacznie więcej kodu niż przy opracowywaniu natywnymi narzędzi oraz to że aplikacja na bazie frameworka Xamarin działają wolniej niż zwykła natywna aplikacja. Na podstawie wyższych analiz poszczególnych testów została stworzona podsumowująca tabela 5.1.

Jak widać z rezultatów sumarycznych wag, natywne narzędzie wytwarzania aplikacji Java, ma 80 punków, w ten czas jak narzędzie Xamarin ma 60. Każdy z systemów mógł otrzymać maksymalnie 100 punktów. Składowe testy, których wyniki były na końcu sumowane mają różną wartość użytkową. Dla przykładu badanie z analizą działania bazy danych jest jednym z najważniejszych i można było zyskać tutaj 20 punktów. Z kolei test pracy kamery nie jest, aż tak istotny dlatego maksymalny wynik jaki system mógł uzyskać to 5.

Teza, która zdefiniowana dla danej pracy została podważona Framework Xamarin nie pozwala tworzyć bardziej wydajnych aplikacji, jak ma to miejsce w przypadku zwykłej aplikacji z dodatkiem środowiska wykonania Mono. W niektórych przypadkach środowisko to wymaga jednak czasu procesora, z czego wynika wolniejsze działania aplikacji, co zostało udowodnione w danej pracy.

Tabela 1. Podsumowanie punktów poszczególnych testów

Nazwa testu	Punkty		
	Xamarin	Java	Wartość maksymalna
Analiza ilości kodu	5	15	20
Szybkość generowania interfejsu użytkownika	5	15	20
Działanie bazy danych	12	17	20
Pętla z dużą ilością iteracji	18	13	15
Analiza działania kamery	5	5	5
Przetwarzanie plików	15	15	20
Suma wag:	60	80	100

6. Wnioski

W tej pracy było zbadano jedno z narzędzi dla wieloplatformowego wytwarzania oprogramowania. Framework Xamarin na pewno jest wartym uwagi rozwiązaniem dla branży rozwoju oprogramowania mobilnego, ponieważ pozwala tworzyć produkty, nie zagłębiając się w szczegóły konkretnych platform z punktu widzenia wytwarzania oprogramowania.

Po realizacji badania i po otrzymaniu wyników można wyciągnąć kilka wniosków na temat możliwości zastosowania tego frameworka. Z punktu widzenia programowego interfejsu, który zapewnia jednolity dostęp do wszystkich głównych platform programowania, jest to doskonałe narzędzie, który ma szeroki wachlarz możliwości, dokumentację i dużą ilość materiałów. Niemniej jednak warto zauważyć, że ten framework okazał się gorszy z punktu widzenia wydajności w odnoszeniu do natywnych narzędzi działania aplikacji. Sądząc po wynikach, można powiedzieć, że w czasie działania aplikacji na bazie frameworka Xamarin może czasami dość znacznie ustępować natywnym narzędzi, co może być istotne dla niektórych aplikacji.

Można powiedzieć, że zastosowanie tego frameworka tak samo może się różnić w zależności od wymagań aplikacji. Jeśli aplikacja nie pracuje na dużej ilości danych i nie wymaga dynamicznego generowania złożonego interfejsu użytkownika, to ten framework może być zastosowany do realizacji aplikacji mobilnych. W pozostałych przypadkach, framework ten nie jest znacznie gorszy od natywnego środowiska uruchomieniowego. Warto również zauważyć, że przy wszystkich wadach badanego frameworku opisanych wyżej, koszt wytwarzania aplikacji za pomocą narzędzia Xamarin jest znacznie mniejszy.

Literatura

[1] M. Reynolds, Xamarin Mobile Application Development for Android, Packt Publishing Ltd, 2014.

[2] D. Hermes, Xamarin Mobile Application Development: Cross-Platform C# and Xamarin.Forms Fundamentals, Apress, 2015.

[3] M. Reynolds, Xamarin Essentials, Packt Publishing Ltd, 2014.

[4] R. Meier, Professional Android 4 Application Development, John Wiley & Sons, 2012.

[5] M. Dave, A. Grant and S. Komatineni, Pro Android 5, Apress, 2015.

[6] Z. Mednieks, L. Dornin, M. Nakamura and Blake Meike, Programming Android, 2nd Edition, O'Reilly Media, Incorporated, 2012.

[7] J. Masner, P. Šimek, J. Jarolímek and I. Hrbek, "Mobile Applications for Agricultural Online Portals – Cross-platform or Native Development," June 2015. [Online]. Available: https://www.researchgate.net/publication/283878658_Mobile_Applications_for_Agricultural_Online_Portals_-_Cross-platform_or_Native_Development. [04.06.2017].

[8] V. Tunalı and S. Z. Erdogan, "Comparison of Popular Cross-Platform Mobile Application Development Tools," October 2015. [Online]. Available: https://www.researchgate.net/publication/282816272_Comparison_of_Popular_Cross-Platform_Mobile_Application_Development_Tools. [04.06.2017].

[9] D. Hermes, "Mobile Development Using Xamarin," January 2015. [Online]. Available: https://www.researchgate.net/publication/300266365_Mobile_Development_Using_Xamarin. [04.06.2017].

[10] L. A. B. Prieto, Z. Komínkova-Oplatková, R. T. Frías and J. L. E. Hernández, "A time performance comparison of particle swarm optimization in mobile devices," January 2016. [Online]. Available: https://www.researchgate.net/publication/309361796_A_time_performance_comparison_of_particle_swarm_optimization_in_mobile_devices. [04.06.2017].

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 213-219
Wysłane: 2017-06-07
Przyjęte: 2017-06-16

Analiza możliwości wykorzystania bibliotek Qt w wybranych zagadnieniach komunikacji międzyprocesowej

Łukasz Janusz Górniak*, Maciej Pańczyk
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. W artykule scharakteryzowano czym jest Qt. Przedstawiono przykładową implementację związaną z komunikacją międzyprocesową wykorzystującą pamięć współdzieloną w klasycznym problemie producenta i konsumenta. Takie rozwiązanie jest mile widziane przez programistów, którzy z różnych powodów piszą swoje programy stosując framework Qt w możliwie szerokim zakresie.

Słowa kluczowe: komunikacja międzyprocesowa; C++; Qt; pamięć współdzielona

*Autor do korespondencji.

Adres/adresy e-mail: lukoog125@wp.pl

Analysis of Qt libraries usage in selected interprocess communication aplications

Łukasz Janusz Górniak*, Maciej Pańczyk
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. This article illustrates what Qt is. An example of the producer-consumer problem implementation in interprocess communication based on the shared memory is presented. Such solution can simplify the process of building interprocess application for those who want to write their programs using Qt in all possible aspects.

Keywords: inter-process communication; C++; Qt; shared memory

*Corresponding author.

E-mail address/addresses: lukoog125@wp.pl

1. Wstęp

Mechanizmy komunikacji międzyprocesowej najczęściej implementuje się z wykorzystaniem bibliotek systemowych [1]. Biblioteki systemowe są zazwyczaj wykonane w języku C [2]. Język C nie należy do najprostszych, zarówno jego nauka jak i programowanie w nim jest trudne i czasochłonne. W obecnych czasach zaś najdroższym elementem w procesie wytwarzania oprogramowania jest czas programisty, dlatego też powstają nowe języki wysokiego poziomu oraz różnego rodzaju biblioteki i frameworki, udostępniające wiele mechanizmów i komponentów wpływających znacznie na szybkość powstawania oprogramowania.

W ramach tego artykułu, przedstawiono alternatywę dla bibliotek systemowych, która ułatwi i przyspieszy proces implementacji komunikacji międzyprocesowej, a za razem pozwoli również na zbudowanie w pełni funkcjonalnego oprogramowania - z graficznym interfejsem użytkownika. Głównym celem jest analiza środowiska Qt oraz przedstawianie przykładowej implementacji komunikacji międzyprocesowej w celu oceny czy Qt spełnia wyżej zdefiniowane założenia. Qt umożliwia implementacje komunikacji międzyprocesowej kilkoma metodami - w artykule ocenie zostanie poddana metoda pamięci dzielonej [3].

2. Czym jest Qt

W tym rozdziale zostały przedstawione podstawowe informacje na temat środowiska Qt. Scharakteryzowano jego budowę, zastosowania oraz opisano specyficzne mechanizmy, które dobrze obrazują jakie ułatwienia zostały wprowadzone w stosunku do "czystego" języka C++. Pozwala to na określenie jakie zalety i wady niesie ze sobą korzystanie z Qt.

2.1. Historia i definicja Qt

Twórcami i pomysłodawcami Qt byli Haavard Nord i Eirik Chambe-Eng. Pierwsze prace nad biblioteką rozpoczęły się w 1991 roku. Aktualna wersja biblioteki to 5.8, wydana 23 stycznia 2017 roku [4].

Qt jest to zestaw bibliotek i narzędzi programistycznych przeznaczonych głównie dla języka C++ i QML (deklaratywnego język programowania opartego na JavaScript, służącego przede wszystkim do tworzenia graficznego interfejsu użytkownika), wspierających wieloplatformowość [5]. Pozwala na pisanie aplikacji desktopowych i mobilnych oraz dla systemów wbudowanych. W Qt można tworzyć proste aplikacje konsolowe, aplikacje z zaawansowanym graficznie interfejsem użytkownika i serwery. Qt może także zostać wykorzystane do budowania oprogramowania związanego z komunikacją międzyprocesową. Najpopularniejszymi wspieranymi

platformami są Linux, Windows, Mac OS X, Android i iOS [6].

Qt można zainstalować korzystając z instalatora w wersji online lub offline. Qt jest dostępne zarówno na licencjach wolnego oprogramowania jak i licencji komercyjnej [7].

2.2. Środowisko programistyczne

Jednym z głównych narzędzi, udostępnianych przez Qt jest zintegrowane środowisko programistyczne nazwane Qt Creator [8]. Qt Creator może być wykorzystany do nauki i tworzenia małych projektów, przez osoby rozpoczynające przygodę z Qt jak i do rozwijania dużych i zaawansowanych programów, przez doświadczonych programistów.

Tak jak w innych tego typu narzędziach, podstawowymi funkcjonalnościami są: możliwość utworzenia i skonfigurowania projektu, edycji plików oraz łatwa praca z kompilatorami. Qt Creator dostarcza rozbudowany edytor kodu który dba o poprawną składnie kodu, kolorowanie kodu, podkreślanie błędów, generowanie kodu, formatowanie kodu czy wyszukiwanie danej frazy w kodzie. W przypadku aplikacji z graficznym interfejsem użytkownika, można skorzystać z wbudowanych edytorów, pozwalających na budowę interfejsu metodą "przeciągnij i upuść". Qt dostarcza dwa tego typu edytory w zależności czy interfejs graficzny jest budowany w QML (nowoczesny wygląd) czy z wykorzystaniem Qt Widgets (tradycyjny wygląd). Z poziomu edytora można zdefiniować powiązania pomiędzy obiektami, począwszy od układu, po oddziaływanie na siebie (mechanizm sygnałów i slotów), jak i określić ich właściwości np. nazwę, wielkość, kolor, marginesy, czcionki i wiele innych [8].

Qt Creator współpracuje również z różnymi zewnętrznymi narzędziami [9]. Jednym z takich narzędzi jest debuger. Środowisko wspiera wiele różnych debugerów przy pomocy których można łatwo śledzić prace aplikacji, wyniki są prezentowane w postaci wizualnej. Kolejnymi wspieranymi narzędziami są systemy kontroli wersji, narzędzia do porównywania kodu oraz symulatory urządzeń np. do zastosowań w przypadku rozwijania aplikacji mobilnych.

2.3. Mechanizmy w Qt

C++ to obiektowy język programowania, zaprojektowany przez Bjarne Stroustrupa, jako rozszerzenie języka C. Charakteryzuje się wysoką wydajnością, jest językiem ogólnego przeznaczenia. Programowanie aplikacji w tym języku zazwyczaj wymaga korzystania z niestandardowych bibliotek. Qt można nazwać pewnego rodzaju nakładką na C++, która ma na celu ułatwienie i przyspieszenie wytwarzania oprogramowania. Aby osiągnąć taki rezultat, stosuje się różnego rodzaju kombinacje makr, sztuczki z preprocesorem i dodatkowe narzędzia, a więc stylu programowania używanego w aplikacjach Qt nie można określić mianem "czystego" C++. Pomimo wprowadzonych usprawnień kod napisany w Qt może być swobodnie łączony z kodem w C/C++.

Qt dostarcza kilka specyficznych mechanizmów, które zwiększają możliwości bazowego języka. Implementacja większości z tych mechanizmów, opiera sie na specjalnej klasie o nazwie QObject. W tym rozdziale opisano trzy podstawowe mechanizmy.

2.3.1. Meta-objekty (Meta-Object System)

Metaobiektem określa się obiekt, który opisuje strukturę innego obiektu. System meta-objektów zaimplementowany w Qt to mechanizm który pozwala na:

- uzyskiwanie informacji o faktycznym typie obiektu, podczas wykonywania programu - RTTI (Run Time Type Information), nawet jeśli programista dysponuje tylko wskaźnikiem na klasę bazową,
- komunikację miedzy obiektami - mechanizm sygnałów i slotów,
- stanowi bazę dla dynamicznego systemu właściwości obiektu (Property System).

Na mechanizm meta-objektów składają się trzy podstawowe elementy [10]:

- QObject - aby korzystać z mechanizmu meta-obiektów, należy dziedziczyć z tej klasy. Do uzyskania podstawowych informacji o obiekcie wykorzystuje się metodę QObject::metaObject() zwracającą obiekt klasy QMetaObject, która udostępnia szerokie API, pozwalające na pozyskanie informacji o danej klasie np. nazwę klasy, ilość oraz szczegóły zdefiniowanych metod, ilość dynamicznych i statycznych właściwości zawartych w klasie oraz szczegóły o każdym z tych elementów, informacje o typach wyliczeniowych i wiele innych. Każdy obiekt może także posiadać własną nazwę - metody setObjectName() i objectName(). Qt udostępnia, do identyfikacji typów, oprócz standardowych operatorów z języka C++ (dynamic_cast oraz typeid) dodatkowy operator qobject_cast oraz metodę QObject::inherits(), która przyjmuje jako parametr nazwę klasy,
- Q_OBJECT - dziedziczenie z QObject nie jest jedynym wymaganiem jakie musi być zrealizowane w celu korzystania z systemu meta-obiektów. Kolejnym wymaganiem jest umieszczenie specjalnego makra w sekcji prywatnej danej klasy. Makro Q_OBJECT definiuje dodatkowe metody i pola, które pozwalają na korzystanie z mechanizmu meta-obiektów oraz z mechanizmu sygnałów i slotów,
- Meta-Object Compiler (moc) - jest to specjalny kompilator, którego zadaniem jest generowanie plików implementacyjnych dla klas, które zawierają makro Q_OBJECT. Pliki zawierają implementacje mechanizmu meta-objektów dla tych klas.

2.3.2. Sygnały i sloty (Signals and Slots)

Mechanizm sygnałów i slotów bazuje na systemie meta-objektów [11]. Jest to rozszerzenie pozwalające na łatwą komunikacje pomiędzy obiektami. Ten mechanizm to implementacja wzorca projektowego o nazwie obserwator (inaczej nazywanego Wydawca-Prenumerator) [12].

Sygnały i sloty są wykorzystywane zarówno w aplikacjach z graficznym interfejsem użytkownika jak i w programach konsolowych. Przykładowo sygnał może odpowiadać zdarzeniu kliknięcia na przycisk, a slot reakcji na to zdarzenie. Do powiązania sygnału ze slotem wykorzystuje sie metodę QObject::connect(). Sygnały można łączyć zarówno z slotami jak i z innymi sygnałami. Mechanizm ten wymaga, aby w aplikacji była uruchomiona pętla zdarzeń (mechanizm przetwarzający zdarzenia) [8].

2.3.3. Relacje pomiędzy obiektami (Object Trees)

Kolejnym mechanizmem jaki dostarcza Qt jest możliwość układania obiektów w strukturę drzewa. Jest to implementacja wzorca projektowego o nazwie kompozyt [12]. Taka relacja (rodzic-dziecko) pozwala na łatwiejsze zarządzanie obiektami.

Klasy zbudowane w oparciu o system meta-objektów udostępniają metodę QObject::setParent() oraz QObject::parent(), które pozwalają na ustawienie rodzica dla danego obiektu oraz pobranie rodzica tego obiektu. Kolejną udostępnianą metoda jest metoda children(), która zwraca wszystkie dzieci danego obiektu.

Założenie jest takie, że jeden obiekt może mieć jednego rodzica i wiele dzieci. Takie podejście pozwala na łatwe przeglądanie struktury obiektów, jakie zawiera dany obiekt (rodzic), co pozwala na automatyzacje procesu zwalniania pamięci - Qt podchodzi do tego w ten sposób, że w momencie niszczenia obiektu rodzica, są automatycznie niszczone obiekty dzieci [13].

3. Przykłady komunikacji międzyprocesowej z zastosowaniem Qt

Qt jest zbudowane z wielu modułów i ma szeroki zakres zastosowań, umożliwia tworzenie rozbudowanych graficznie aplikacji oraz zaawansowanych programów konsolowych. Dostarcza również kilka dróg pozwalających na implementacje komunikacji międzyprocesowej. W tym przykładzie skupiono się na metodzie pamięci dzielonej.

Jednym z problemów, jakie spotyka się implementując przesyłanie danych pomiędzy współpracującymi ze sobą współbieżnymi procesami jest "problem producenta i konsumenta". Problem producenta i konsumenta należy do grupy klasycznych informatycznych problemów synchronizacji. Podstawowe założenia tego problemu wyglądają następująco:

- jeden proces (producent) jest odpowiedzialny za generowanie (produkcje) danych, natomiast zadaniem drugiego (konsument) jest odbiór danych (produktów),
- wytworzone produkty są umieszczane w magazynie,
- nie można umieszczać produktów w pełnym magazynie,
- nie można pobierać produktów z pustego magazynu.

W ramach analizy wykonano opis klasy wykorzystanej w przykładzie, dokonano przeglądu interfejsu tej klasy oraz przedstawiono przykładową implementacje rozwiązującą wyżej zdefiniowany problem.

3.1. Opis dostarczonych klas i ich interfejsów

Jednym z najszybszych sposobów komunikacji międzyprocesowej jest pamięć współdzielona. Qt umożliwia użycie tej metody komunikacji wykorzystując klasę QSharedMemory [14]. Klasa QSharedMemory została zaprojektowana w celu stworzenia prostego interfejsu pozwalającego na implementację wymiany danych poprzez segment pamięci współdzielonej. Dodatkowym usprawnieniem jest wbudowany mechanizm zapewniający spójność danych, wykorzystujący semafory [1].

W tabeli 1, został przedstawiony opis najważniejszych metod jakie są dostępne do dyspozycji programisty. Klasa udostępnia również dwa typy wyliczeniowe: AccessMode oraz SharedMemoryError. Typ AccessMode opisuje tryby dostępu do pamięci. Dostęp do pamięci może być realizowany na dwa sposoby: ReadWrite oraz ReadOnly. Domyślnym trybem jest ReadWrite, który daje możliwość zapisu danych do pamięci jak i odczyt z pamięci. Natomiast typ wyliczeniowy o nazwie SharedMemoryError dostarcza listę błędów, które mogą wystąpić podczas wykonywania programu.

3.2. Badania

W ramach badań przygotowano dwie aplikacje, które rozwiązują problem producenta i konsumenta. Założono, że producent ma za zadanie wyprodukowanie 200000 produktów i dysponuje magazynem o ograniczonej wielkości 4096 bajtów.

Aplikacje składają się z dwóch głównych klas: Producent oraz Konsument. Obie klasy implementują metodę obsluzBlad(), jest to metoda o bardzo prostej logice - w przypadku wystąpienia błędu wysyła komunikat na wyjście i kończy działanie aplikacji. Metoda przyjmuje dwa argumenty, pierwszy określający czy wystąpił błąd, drugi zawierający treść komunikatu, jaki ma być wyświetlony w przypadku błędu.

Listing 1 przedstawia interfejs klasy Producent. W klasie została zadeklarowana metoda start(), która rozpoczyna "produkcje" oraz metoda losujLiczbe(), pozwalająca na generowanie pseudolosowych liczb z zadanego zakresu (produktów), a także wspomniana wyżej metoda obsluzBlad(). W sekcji prywatnej umieszczono deklaracje typów: sc_klucz jest to zmienna przechowująca klucz do pamięci współdzielonej w postaci łańcucha znaków, sc_maxIloscProduktow i sc_maxRozmiarPamieci określają kolejno ilość produktów do wyprodukowania oraz maksymalny rozmiar danych jakie mogą zostać umieszczone w pamięci, m_pamiecDzielona to wskaźnik na obiekt wystawiający interfejs do operowania na segmentach pamięci współdzielonej, zmienna m_wyprodukowano jest wykorzystywana do zliczania wyprodukowanych produktów, natomiast obiekt m_dane gromadzi wyprodukowane produkty (liczby) w postaci tablicy bajtów.

Tabela 1. Opis metod klasy QSharedMemory [14].

Typ zwracany	Nazwa metody	Opis
	QSharedMemory(const QString &key, QObject *parent = Q_NULLPTR)	Konstruuje obiekt o zadanym kluczu i rodzicu. Aby utworzyć segment pamięci współdzielonej należy wywołać metodę create(). W celu dołączenia do już istniejącego segmentu, należy wywołać metodę attach().
	~QSharedMemory()	W destruktorze następuje usunięcie klucza oraz odłączenie od pamięci współdzielonej. Jeśli to był ostatni z obiektów połączonych z pamięcią współdzieloną, następuje usunięcie segmentu pamięci współdzielonej.
bool	attach(AccessMode mode = ReadWrite)	Metoda podejmuje próbę dowiązania do segmentu pamięci współdzielonej. Dostęp do pamięci może być realizowany w różnych trybach - domyślnym trybem jest ReadWrite. Gdy operacja się powiedzie, jest zwracane wartość true, w przypadku niepowodzenia wartość false. Metoda error() może zostać wykorzystana do sprawdzenia błędów jakie wystąpiły.
Bool	create(int size, AccessMode mode = ReadWrite)	Wykorzystywana do utworzenia segmentu pamięci współdzielonej. Pierwszy przekazywany parametr określa wielkość segmentu pamięci w bajtach, drugi tryb dostępu. Segment pamięci jest tworzony w oparciu o klucz przekazany w konstruktorze klasy lub ustawiony za pomocą metod setKey() lub setNativeKey(). Jeżeli istnieje pamięć współdzielona o zadanym kluczy, jest zwracane false. Metoda error() może zostać wykorzystana do sprawdzenia błędów jakie wystąpiły.
void *	data()	Metoda zwraca wskaźnik na zawartość pamięci współdzielonej. W przypadku gdy segment pamięci nie istnieje jest zwracany null. Przed operacjami zapisu lub odczytu z pamięci, należy wykorzystać metody lock() lub unlock() w celu synchronizacji dostępu.
Bool	detach()	Odłącza obiekt od pamięci współdzielonej, zwraca true w przypadku powodzenia lub false jeśli operacja się nie powiodła lub jeśli obiekt nie był dowiązany. Wartość false może być także zwrócona w przypadku gdy nie nastąpiło odłączenie ponieważ segment pamięci był zablokowany przez inny proces.
SharedMemoryError	error() const	Zwraca numer błędu opisanego za pomocą enum. W przypadku niewystąpienia błędu jest zwracana wartość QSharedMemory::NoError.
QString	errorString() const	Zwraca opis błędu w postaci łańcucha znaków.
QString	key() const	Zwraca klucz do pamięci współdzielonej, który został przypisany w konstruktorze lub za pomocą metody setKey(). Tego typu klucz jest specyficzny dla aplikacji zaimplementowanych w Qt i rożni się od natywnego klucza nativeKey(), wykorzystywanego przez system. Metoda zwraca pusty łańcuch znaków w przypadku gdy klucz nie został ustawiony lub użyto natywnego klucza.
Bool	lock()	Metoda wykorzystywana do synchronizacji dostępu do segmentu pamięci współdzielonej, korzystająca z semaforów. Zwraca wartość true, gdy udało się zablokować segmentu pamięci na wyłączność obecnego procesu. W przypadku gdy inny proces blokuje dostęp do segmentu, wywołanie powoduje zablokowanie bieżącego procesu do momentu zwolnienia blokady przez proces blokujący i nałożenia jej przez proces bieżący. Wartość false jest zwracana w przypadku stosowania natywnego klucza setNativeKey() lub jeśli nie udało się użyć semafora z powodu błędu systemowego.
QString	nativeKey() const	Zwraca natywny, specyficzny dla platformy klucz, ustawiony za pomocą metody setNativeKey(). Natywny klucz może być wykorzystywany do dowiązania do segmentu pamięci współdzielonej, zarówno przez aplikacje zaimplementowane w Qt jak i przez aplikacje wykonane w innych technologiach. W przypadku gdy klucz został ustawiony za pomocą setKey(), metoda zwraca przekonwertowany klucz do postaci klucza specyficznego dla platformy.
Void	setKey(const QString &key)	Pozwala na ustawienie klucza do pamięci współdzielonej, specyficznego dla aplikacji zaimplementowanych w Qt. Tego rodzaju klucz jest niezależny od platformy. W przypadku gdy następuje zmiana klucza, a obiekt jest dowiązany do segmentu pamięci współdzielonej, następuje jego odwiązanie. Aby ponownie dowiązać obiekt należy skorzystać z metody attach().
Void	setNativeKey(const QString &key)	Ustawia klucz do pamięci współdzielonej, specyficzny dla danej platformy. W przypadku gdy nastąpi zmiana klucza, a obiekt jest dowiązany do segmentu pamięci współdzielonej obiekt zostanie automatycznie odwiązany. Aby ponownie dowiązać obiekt należy skorzystać z metody attach(). Użycie tego rodzaju klucza sprawia, że aplikacja nie będzie wieloplatformowa.
Int	size() const	Metoda zwraca wielkość (w bajtach) segmentu pamięci współdzielonej, do którego jest dowiązany obiekt, w przypadku gdy obiekt nie jest dowiązany jest zwracana wartość 0.
Bool	unlock()	Jeżeli blokada na segment pamięci współdzielonej została założona przez ten proces, następuję odblokowanie i zwrócenie wartości true. Jeżeli blokada nie została założona lub jeśli blokadę założył inny proces, metoda zwraca wartość false.

Listing 1. Plik nagłówkowy klasy Producent

```
1 #include <QObject>
2 class QSharedMemory;
3 class Producent: public QObject
4 {
5     Q_OBJECT
6 public:
7     explicit Producent (QObject *rodzic = 0);
8
9 public slots:
10     void start();
11
12 private:
13     static const QString sc_klucz;
14     static const int sc_maxIloscProduktow;
15     static const int sc_maxRozmiarPamieci;
16     QSharedMemory *m_pamiecDzielona;
17     int m_wyprodukowano = 0;
18     QByteArray m_dane;
19     int losujLiczbe(int min, int max);
20     bool obsluzBlad(bool blad, const QString &tresc);
21 };
```

Listing 2 przedstawia plik źródłowy klasy Producent. W linijkach 8-11 znajdują się definicje stałych, w konstruktorze klasy jest tworzony obiekt klasy QSharedMemory i ustawiany klucz na wartość "producent-konsument-klucz". Na początku metody start() jest tworzony segment pamięci o wskazanej wielkości przy pomocy metody create(), jeżeli segment pamięci został utworzony poprawnie, zostaje wypisany na wyjście klucz, natomiast w przypadku niepowodzenia komunikat o błędzie. Niepowodzenie również skutkuje zakończeniem aplikacji (linia 20). W linijkach 31-32 jest pobierany czas, który zostaje wykorzystany, aby zainicjować generator liczb pseudolosowych wykorzystywany przez metodę losujLiczbe(). Linijki 35-71 zawierają główną pętle aplikacji, na początku pętli następuje próba zablokowania segmentu pamięci na wyłączność danego procesu (linia 36), w przypadku gdy inny proces blokuje dostęp do segmentu, wywołanie powoduje zablokowanie bieżącego procesu do momentu zwolnienia blokady przez proces blokujący i nałożenia jej przez proces bieżący. Po nałożeniu blokady następuje pobranie zawartości pamięci współdzielonej i sprawdzenie jej rozmiaru. Jeśli pamięć jest pełna blokada jest opuszczana i program przechodzi do ponownego rozpoczęcia pętli, co daje czas innemu procesowi (konsumentowi) na dostęp do pamięci. W przypadku gdy rozmiar danych jest mniejszy od maksymalnie dopuszczalnego, następuje przystąpienie do wyprodukowania produktu i umieszczenia go w pamięci. Jest to realizowane w linijkach 51-56, na początku aplikacja losuje liczbę i dopisuję ją na końcu bufora, następnie zawartość bufora zostaje skopiowana do segmentu pamięci współdzielonej oraz następuje opuszczenie blokady. Linijki 58-59 odpowiadają za aktualizacje stanu produkcji - inkrementacje ilości wyprodukowanych produktów oraz dodanie kolejnego produktu (liczby) do tablicy bajtów, gdzie znajdują się kolejno wyprodukowane produkty. Linijki 61-70 są odpowiedzialne za przerwanie wykonywania programu gdy zostanie osiągnięty limit produkcji, w tym przypadku zostają wypisane na wyjście: informacja o ilości wyprodukowanych produktów, funkcja skrótu wyliczona dla zgromadzonych danych (MD5) oraz komunikat o zakończeniu.

Listing 2. Plik źródłowy klasy Producent

```
1 #include "producent.h"
2 #include <QCoreApplication>
3 #include <QSharedMemory>
4 #include <QDebug>
5 #include <QTime>
6 #include <QCryptographicHash>
7
8 const QString Producent::sc_klucz = "producent-
9 konsument-klucz";
10 const int Producent::sc_maxIloscProduktow = 200000;
11 const int Producent::sc_maxRozmiarPamieci = 4096;
12
13 Producent::Producent(QObject *rodzic)
14   : QObject(rodzic),
15    m_pamiecDzielona{new QSharedMemory(sc_klucz,
16 this)} {}
17
18 void Producent::start()
19 {
20   if (obsluzBlad(!m_pamiecDzielona-
21 >create(sc_maxRozmiarPamieci),
22                  "Utworzenie segmentu pamieci
23 wspoldzielonej nie bylo mozliwe:")) {
24       return;
25    } else {
26       qDebug () << "Utworzono segment pamieci
27 wspoldzielonej o kluczu:"
28                 << m_pamiecDzielona->key();
29   }
30    QByteArray bufor;
31    QTime czas = QTime::currentTime();
32    qsrand((uint)czas.msec());
33    int produkt = 5;
34
35   while (true) {
36       if (obsluzBlad(!m_pamiecDzielona->lock(),
37 "Zakladanie blokady nie powiodlo sie:"))
38           return;
39
40       char *zawartosc = (char*)m_pamiecDzielona-
41 >data();
42       bufor = zawartosc;
43       if (bufor.size() + 1 >= sc_maxRozmiarPamieci) {
44           if (obsluzBlad(! m_pamiecDzielona->unlock(),
45 "Zdejmowanie blokady nie powiodlo sie:"))
46               return;
47
48           continue;
49       }
50
51       produkt = losujLiczbe(1, 9);
52       bufor.append(QByteArray::number(produkt));
53       strcpy(zawartosc, bufor.constData());
54
55       if (obsluzBlad(! m_pamiecDzielona->unlock(),
56 "Zdejmowanie blokady nie powiodlo sie:"))
57           return;
58       ++m_wyprodukowano;
59      m_dane.append(QByteArray::number(produkt));
60
61       if (m_wyprodukowano == sc_maxIloscProduktow)
62
63           qDebug () << "Wyprodukowano" <<
64 m_wyprodukowano << "produktow.";
65           qDebug () << "MD5:" <<
66 QCryptographicHash::hash(m_dane,
67 QCryptographicHash::Md5).toHex();
68           qDebug () << "Zakonczono.";
```

```
69            QCoreApplication::exit(0);
70            return;
71        }
72    }
73 }
74
75 int Producent::losujLiczbe(int min, int max)
76 {
77     return qrand() % ((max + 1) - min) + min;
78 }
```

Interfejs klasy Konsument, jest bardzo podobny do interfejsu klasy Producent. W klasie została zadeklarowana metoda start(), która rozpoczyna pobieranie danych oraz metoda obsluzBlad(), do prostej obsługi błędów. W sekcji prywatnej umieszczono deklaracje typów: sc_klucz jest to zmienna przechowująca klucz do pamięci współdzielonej w postaci łańcucha znaków, sc_maxIloscProduktow określa ilość produktów jakie mają zostać pobrane, m_pamiecDzielona to wskaźnik na obiekt wystawiający interfejs do operowania na segmentach pamięci współdzielonej, zmienna m_pobrano jest wykorzystywana do zliczania pobranych produktów, natomiast obiekt m_dane gromadzi pobrane produkty (liczby) w postaci tablicy bajtów.

Listing 3 przedstawia plik źródłowy klasy Konsument. Stałe oraz klucz do pamięci współdzielonej są takie same jak w klasie Producent. W metodzie start() jest podejmowana próba dowiązania do segmentu pamięci współdzielonej przy pomocy wywołania metody attach(), jeżeli segment pamięci o danym kluczu istnieje i udało się dowiązać zostaje wypisany na wyjście klucz, natomiast w przypadku niepowodzenia nastąpi wypisanie komunikatu o błędzie i zakończenie wykonywania aplikacji (linia 18). Linijki 27-58 zawierają główną pętle aplikacji, podobnie jak w klasie Producent, na początku pętli następuje próba zablokowania segmentu pamięci na wyłączność danego procesu, metodą lock() (linia 28), w przypadku gdy inny proces blokuje dostęp do pamięci, wywołanie powoduje zablokowanie bieżącego procesu do momentu zwolnienia blokady przez proces blokujący i nałożenia jej przez proces bieżący. Po nałożeniu blokady, następuje pobranie zawartości pamięci współdzielonej do bufora i sprawdzenie jej rozmiaru. Jeśli pamięć jest pusta, blokada jest opuszczana i program przechodzi do ponownego rozpoczęcia pętli, co daje czas procesowi producenta na umieszczenie w magazynie nowych produktów. W przypadku gdy rozmiar danych jest większy od 0, pamięć współdzielona zostaje wyczyszczona, po wyczyszczeniu następuje opuszczenie blokady (linijki 41-44). Linijki 45-46 odpowiadają za aktualizacje stanu przetwarzania - zapisanie danych z bufora do tablicy bajtów, która przechowuje pobrane produkty oraz zwiększenie liczby pobranych elementów o wielkość bufora. Linijki 48-57 są odpowiedzialne za zakończenie programu gdy zostanie osiągnięty limit pobranych danych, w tym przypadku zostają wypisane na wyjście: informacja o ilości pobranych produktów, funkcja skrótu wyliczona dla przechowywanych danych (MD5) oraz komunikat o zakończeniu.

Listing 3. Plik źródłowy klasy Konsument

```
1 #include "konsument.h"
2 #include <QCoreApplication>
3 #include <QSharedMemory>
4 #include <QDebug>
5 #include <QCryptographicHash>
6
7 const QString Konsument::sc_klucz = "producent-
8 konsument-klucz";
9 const int Konsument::sc_maxIloscProduktow = 200000;
10
11 Konsument::Konsument(QObject *rodzic)
12   : QObject(rodzic),
13    m_pamiecDzielona{new QSharedMemory(sc_klucz,
14 rodzic)} {}
15
16 void Konsument::start() {
17    if (obsluzBlad(!m_pamiecDzielona->attach(),
18                "Dowiazanie do segmentu pamieci
19 wspoldzielonej nie bylo mozliwe:")) {
20        return;
21    } else {
22        qDebug () << "Dowiazano segment pamieci
23                wspoldzielonej o kluczu:"
24                << m_pamiecDzielona->key();
25    }
26    QByteArray bufor;
27   while (true) {
28        if (obsluzBlad(!m_pamiecDzielona->lock(),
29 "Zakladanie blokady nie powiodlo sie:"))
30            return;
31        char *zawartosc = (char*)m_pamiecDzielona-
32 >data();
33        bufor = zawartosc;
34        if (!bufor.size()) {
35            if (obsluzBlad(! m_pamiecDzielona->unlock(),
36                "Zdejmowanie blokady nie powiodlo sie:"))
37                return;
38            continue;
39        }
40
41        zawartosc[0] = 0;
42        if (obsluzBlad(! m_pamiecDzielona->unlock(),
43            "Zdejmowanie blokady nie powiodlo sie:"))
44            return;
45        m_dane.append(bufor);
46        m_pobrano += bufor.size();
47
48        if (m_pobrano == sc_maxIloscProduktow) {
49            qDebug () << "Pobrano" << m_pobrano <<
50                        "produktow.";
51            qDebug () << "MD5:" <<
52                QCryptographicHash::hash(m_dane,
53                QCryptographicHash::Md5).toHex();
54            qDebug () << "Zakonczono.";
55            QCoreApplication::exit(0);
56            return;
57        }
58    }
59 }
```

4. Wnioski

Klasa QSharedMemory udostępnia rozbudowany interfejs do pracy z pamięcią współdzieloną, pozwala zarówno na łatwe utworzenie obszaru pamięci współdzielonej o dowolnym rozmiarze, jak i dowiązanie do niego, a także dba o jego usunięcie w momencie zakończenia pracy ostatniego dowiązanego procesu. Do identyfikacji segmentu pamięci można wykorzystać dwa rodzaje kluczy: specyficzny dla Qt oraz specyficzny dla

danej platformy. Takie rozwiązanie pozwala na tworzenie obszaru pamięci współdzielonej przez aplikację zaprogramowaną w Qt i korzystanie z tego obszaru przez aplikacje wykonane w innych technologiach. Interfejs klasy pozwala również na łatwe śledzenie błędów, błędy mogą być pobierane w postaci typu wyliczeniowego lub w postaci łańcucha znaków - krótkiego opisu słownego. Programista może określić, w łatwy sposób, w jakim trybie będzie realizowany dostęp do pamięci - proces może otrzymać prawo do zapisu i odczytu lub wyłącznie do odczytu. Operacja zapisu lub odczytu jest realizowana w stylu języka C, interfejs klasy nie pozwala na łatwy zapis oraz sprawdzenie wielkości zapisanych danych, a więc programista korzystający z tej metody musi znać podstawy języka C. Klasa QSharedMemory posiada wbudowany mechanizm synchronizacji dostępu do pamięci, który okazał sie bardzo prosty w użyciu, podczas implementacji przykładowych aplikacji. Spójność danych została sprawdzona za pomocą funkcji skrótu (MD5), wyliczonej na wygenerowanych i pobranych danych. Wyniki pracy programów zostały przedstawiona na rysunku 1 i 2. Zarówno w aplikacji producenta jak i aplikacji konsumenta otrzymano taki sam skrót.

```
Utworzono segment pamieci wspoldzielonej o kluczu: "producent-konsument-klucz"
Wyprodukowano 200000 produktow.
MD5: "0cd02cf2a1f2c22bccd8229fbc3537f1"
Zakonczono.
```

Rys. 1. Wynik pracy aplikacji Producent.

```
Dowiazano segment pamieci wspoldzielonej o kluczu: "producent-konsument-klucz"
Pobrano 200000 produktow.
MD5: "0cd02cf2a1f2c22bccd8229fbc3537f1"
Zakonczono.
```

Rys. 2. Wynik pracy aplikacji Konsument.

Literatura

[1] J. S. Gray, Komunikacja między procesami w Unixie. ReadMe, Warszawa 1998

[2] John Fusco, The Linux Programmer's Toolbox, Pearson Education, 2007

[3] http://doc.qt.io/qt-5/ipc.html [24.05.2017]

[4] http://wiki.qt.io/Qt_History [24.05.2017]

[5] Symeon Huang, Qt 5 Blueprints, Pack Publishing, 2015

[6] Guillaume Lazar, Robin Penea, Mastering Qt 5, Packt Publishing , 2016

[7] http://doc.qt.io/qt-5/licensing.html [24.05.2017]

[8] A. Ezust, P. Ezust, Qt. Wprowadzenie do wzorców projektowych. Wydanie II, Helion, 2014

[9] http://doc.qt.io/qt-5/topics-app-development.html [24.05.2017]

[10] http://doc.qt.io/qt-5/metaobjects.html [24.05.2017]

[11] http://doc.qt.io/qt-5/signalsandslots.html [24.05.2017]

[12] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku. Helion, 2010

[13] http://doc.qt.io/qt-5/objecttrees.html [24.05.2017]

[14] http://doc.qt.io/qt-5/qsharedmemory.html [24.05.2017]

JOURNAL of COMPUTER SCIENCES INSTITUTE

JCSI 5 (2017) 220-226
Wysłane: 2017-08-28
Przyjęte: 2017-08-31

Ocena stopnia wykorzystania e-usług w administracji samorządowej w województwie lubelskim

Daniel Ziętek*, Elżbieta Miłosz
Politechnika Lubelska, Instytut Informatyki, Nadbystrzycka 36B, 20-618 Lublin, Polska

Streszczenie. Jednostki Samorządu Terytorialnego województwa lubelskiego mają dostęp do systemów i narzędzi umożliwiających tworzenie, udostępnianie oraz procedowanie e-usług dlatego też, powinny udostępniać szereg usług obywatelom i przedsiębiorcom, stale je aktualizować i podnosić poziom ich dojrzałości. W artykule przedstawiono wyniki badań, które obejmowały ocenę stopnia wykorzystania e-usług w administracji samorządowej w województwie lubelskim.

Słowa kluczowe: e-usługi; ESP; ePUAP; LIBOI

*Autor do korespondencji.

Adres e-mail: zietek.daniel@gmail.com

Evaluation of the utilization rate of e-services provided by the local administration in Lubelskie Voivodeship

Daniel Ziętek*, Elżbieta Miłosz
Institute of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract. Local Administration Entities of Lubelskie Voivodeship have access to systems and tools allowing for creating, implementing and providing access to e-services. Thus, they should make these services available to citizens and entrepreneurs, update them on a regular basis and improve their maturity level. This article presents results of research, including evaluation of the utilization rate of e-services provided by the local administration in Lubelskie Voivodeship.

Keywords: e-services; ESP; ePUAP; LIBOI

*Corresponding author.

E-mail address: zietek.daniel@gmail.com

1. Wstęp

Poziom informatyzacji województwa lubelskiego w dalszym ciągu pozostaje niewystarczający oraz odbiega od poziomu wykorzystania technologii informatycznych w realizowaniu usług przez inne regiony w kraju oraz za granicą. Zarówno infrastruktura sprzętowa jak i usługi elektroniczne w dużym stopniu wymagają nowych nakładów inwestycyjnych, aby móc sprostać wymaganiom stawianym przez klientów urzędów.

Głównym problemem wpływającym na niski poziom usług świadczonych drogą elektroniczną jest mała liczba udostępnianych w ten sposób usług jak również niski poziom dojrzałości tych, które są dostępne. Narzędziem służącym do komunikacji obywatela z urzędem jest Elektroniczna Skrzynka Podawcza, do posiadania której zobligowane są wszystkie Jednostki Samorządu Terytorialnego (JST). Jednak poziom wykorzystania tego narzędzia jest bardzo zróżnicowany.

2. E-usługi

Istnieje wiele definicji e-usług. Wynika to faktu, że nie można jednoznacznie wskazać granic e-usługi ze względu na dynamiczny rozwój środowiska, w którym się ona tworzy i kształtuje. Konsekwencją tego jest problem związany ze wskazaniem czy konkretna aktywność internetowa zaliczana jest do e-usługi czy też nie.

Jedną z definicji e-usługi zawiera Ustawa z dnia 18 lipca 2002 r. o świadczeniu usług drogą elektroniczną. Definiuje ona świadczenie usługi drogą elektroniczną jako „*wykonanie usługi świadczonej bez jednoczesnej obecności stron (na odległość), poprzez przekaz danych na indywidualne żądanie usługobiorcy, przesyłanej i otrzymywanej za pomocą urządzeń do elektronicznego przetwarzania, włącznie z kompresją cyfrową i przechowywania danych, która jest w całości nadawana, odbierana lub transmitowana za pomocą sieci telekomunikacyjnej w rozumieniu ustawy z dnia 16 lipca 2004 r. – Prawo telekomunikacyjne”*.[1]

Zbliżoną definicję podaje również Rozporządzenie Rady Unii Europejskiej (WE) nr 1777/2005 z dnia 17 października 2005 r. ustanawiające środki wykonawcze do dyrektywy 77/388/EWG w sprawie wspólnego systemu podatku od wartości dodanej, której art. nr 11 ust. 1 mówi, że „*do usług świadczonych drogą elektroniczną, należą usługi świadczone za pomocą Internetu lub sieci elektronicznej, których świadczenie jest zautomatyzowane i które wymagają*

niewielkiego udziału człowieka, a ich wykonanie bez wykorzystania technologii informacyjnej jest niemożliwe."[2]

Podsumowując powyższe e-usługi powinny być świadczone w sposób:

- zautomatyzowany (częściowo lub całkowicie) przez technologię informacyjną,
- za pomocą Internetu,
- realizowany zdalnie,
- bez jednoczesnej obecności stron,
- dostępny o każdej porze.

2.1. Obszary e-usług

Analizując e-usługi można wyróżnić obszary ich wdrożeń. Obecnie e-usługi intensywnie rozwijane są w obszarach takich jak administracja, handel, bankowość, ochrona zdrowia, turystyka, kultura, komunikacja, nauka, praca.

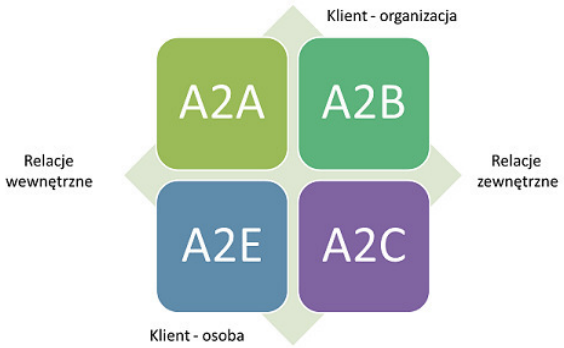
Technologia informacyjna i komunikacyjna w ww. obszarach określana jest jako:

- e-administracja, czyli usprawnienie działań związanych z realizacją zadań publicznych polegających na udostępnieniu e-usług w kanałach elektronicznych,
- e-handel, czyli sprzedaż, kupno produktów i usług za pośrednictwem Internetu np. poprzez aukcje internetowe czy sklepy internetowe,
- e-bankowość, usługi bankowości elektronicznej dostępne poprzez sieć (Internet) w zakresie obsługi np. konta bankowego, kart płatniczych, kredytów,
- e-zdrowie, czyli systemy i usługi wykorzystujące technologię teleinformatyczną wykorzystywaną przez pracowników służb zdrowia, którą można dostosować do indywidualnych potrzeb pacjentów,
- e-turystyka, czyli promocja w Internecie usług i produktów związanych z turystyką,
- e-kultura, czyli udostępnienie dziedzictwa kulturowego w Internecie, do którego zaliczają się między innymi biblioteki cyfrowe, galerie, muzea, e-booki, teatry, strony www poświęcone tematyce kultury,
- e-komunikacja, czyli kontaktowanie się z ludźmi za pomocą Internetu np. za pomocą komunikatorów typu Skype czy Messenger,
- e-nauka, czyli szkolenie na odległość z wykorzystaniem Internetu tzw. e-learning,
- e-praca, czyli wykonywanie obowiązków służbowych poza siedzibą firmy, używając do tego sieci, komputera, telefonu (telepracy).

2.2. Klasyfikacja rodzajowa usług administracji publicznej

Można wyróżnić następujące klasyfikacje rodzajowe usług administracji publicznej (przedstawione na Rys. 1.):

- A2A (ang. Administration to Administration) – usługi wewnątrzadministracyjne. Są to systemy, serwisy www lub różnego rodzaju usługi, które łączą ze sobą np. urzędy marszałkowskie z jednostkami samorządu terytorialnego.
- A2B (ang. Administration To Business) – usługi administracji dla biznesu. Są to systemy, serwisy www lub różnego rodzaju usługi, które łączą ze sobą organizacje rządowe z przedsiębiorstwami. Przykładem mogą być elektroniczne skrzynki podawcze, e-deklaracje podatkowe dla firm.
- A2C (ang. Administration to Citizen) – usługi administracji dla obywateli. Są to systemy, serwisy www lub różnego rodzaju usługi, które łączą ze sobą organizacje rządowe z klientem. Przykładem mogą być elektroniczne skrzynki podawcze lub inne platformy świadczące e-usługi, serwisy umożliwiające kontakt klienta z urzędem.
- A2E (ang. Administration to Employee) – usługi administracji dla pracowników. Są to systemy, serwisy www lub różnego rodzaju usługi, które łączą ze sobą organizacje rządowe z pracownikami administracji publicznej. Przykładem mogą być systemy elektronicznego zarządzania dokumentacją (EZD).



Rys. 1. Klasyfikacja rodzajowa usług administracji publicznej [3]

2.3. Poziomy dojrzałości e-usług w e-administracji

Sposób załatwienia danej sprawy określa poziom dojrzałości usług oferowanych przez daną instytucję.

Skala dojrzałości usług elektronicznych liczy pięć poziomów:

Poziom 1 - informacyjny - oznaczający, udostępnianie przez urzędy informacji na portalach internetowych bez możliwości zainicjowania i załatwienia sprawy drogą elektroniczną przez obywateli i przedsiębiorców.

Poziom 2 - interakcja jednostronna - obywatele i przedsiębiorcy komunikują się z instytucjami administracji publicznej drogą elektroniczną w celu pobrania formularzy ze strony urzędu lecz jest to komunikacja jednostronna ponieważ nie mają możliwości zainicjowania i załatwienia sprawy drogą elektroniczną.

Poziom 3 - interakcja dwustronna – obywatele i przedsiębiorcy mają możliwość zainicjowania sprawy poprzez wypełnienie formularza online dostępnego na stronie urzędu oraz przesłanie go wraz z dokumentami elektronicznymi do urzędu.

Poziom 4 - transakcyjny – obywatele i przedsiębiorcy mogą w pełni załatwić sprawę drogą elektroniczną tzn. wysyłając formularz elektroniczny do urzędu poprzez np. elektroniczną skrzynkę podawczą (ESP) obywatel/przedsiębiorca otrzyma odpowiedź z urzędu również drogą elektroniczną oraz uzyska możliwość opłaty za usługę online w przypadku gdy taka opłata będzie wymagana.

Poziom 5 - personalizacja – oznacza, załatwienie sprawy drogą elektroniczną przy jednoczesnym wprowadzeniu personalizacji obsługi, tzn. obywatel lub przedsiębiorca otrzymuje automatycznie konkretne usługi, spersonalizowane pod niego bez wcześniejszego inicjowania.[4]

3. Narzędzia do udostępniania i tworzenia e-usług

3.1. Elektroniczna Platforma Usług Administracji Publicznej – ePUAP.gov.pl

Definicję Elektronicznej Platforma Usług Administracji Publicznej (ePUAP) zawiera art. 3 pkt 13 ustawy o informatyzacji działalności podmiotów realizujących zadania publiczne, w którym mowa o systemie teleinformatycznym udostępniającym przez Internet usługi instytucji publicznych.[5] Usługi udostępniane są dla osób fizycznych i podmiotów gospodarczych. ePUAP dodatkowo służy również do komunikacji pomiędzy jednostkami administracji publicznej. Usługodawcami są podmioty administracji publicznej oraz inne podmioty wykonujące zadania publiczne zlecone lub powierzone im do realizacji.[6]

3.2. Lubelskie Internetowe Biuro Obsługi Interesanta – LIBOI.lubelskie.pl

Platforma LIBOI spełnia funkcję Elektronicznej Skrzynki Podawczej. Z platformy LIBOI korzystać mogą partnerzy projektu „Wrota Lubelszczyzny – Informatyzacja Administracji" oraz ich jednostki organizacyjne.

Platforma wyposażona została w narzędzie edytor formularzy, który pozwala użytkownikom po stronie JST na samodzielne tworzenie i wystawianie swoich usług w wersji w pełni elektronicznej. System daje możliwość na realizację usług o każdym poziomie dojrzałości (pięciostopniowa skala), w tym możliwość wystawiania formularzy do druku czy po prostu informacji o konkretnej usłudze. Platforma przewiduje również możliwość dokonywania płatności online, więc jednostki mogą wystawiać usługi, które takich płatności wymagają. LIBOI umożliwia podpisanie dokumentów przy pomocy bezpłatnego podpisu elektronicznego - Profilu Zaufanego oraz podpisu elektronicznego. Wszystkie dokumenty przynależne do danego użytkownika znajdują się na indywidualnym koncie petenta. Z tego miejsca można odczytać odpowiedź urzędu oraz przejrzeć dokumenty wysłane i edytowane.[7]

3.3. OBYWATEL.GOV.PL

Lista usług dostępnych dla obywatela i przedsiębiorcy jest bardzo długa. Klient urzędu czasami nie jest w stanie sam odszukać interesującej go usługi na Elektronicznych Skrzynkach Podawczych czy stronach Biuletynów Informacji Publicznej urzędów. Często też zdarza się, że dane usługi są opisane bardzo zawile lub językiem urzędniczym lub prawniczym mało zrozumiałym dla przeciętnego obywatela. Dlatego też Ministerstwo Cyfryzacji udostępniło serwis OBYWATEL.GOV.PL, w którym w przejrzysty sposób prezentuje najpopularniejsze usługi opisane prostym językiem.[8]

W serwisie przy usługach znajduje się lista odpowiedzi na najczęściej zadawane pytania do danej sprawy. Również niektóre usługi posiadają wbudowaną ankietę, która zawiera pytania dotyczące udostępnianej usługi. Dzięki odpowiedziom na pytania obywatel zostaje skierowany do odpowiedniego wniosku i/lub urzędu, w którym może załatwić daną sprawę.

3.4. Profil zaufany (eGO) – pz.gov.pl

Definicję Profilu zaufanego (eGO) zawiera art. 3 pkt 14 ustawy o informatyzacji działalności podmiotów realizujących zadania publiczne, w którym mowa o informacjach identyfikujących i opisujących podmiot lub osobę, która została uwierzytelniona za pośrednictwem certyfikatu kwalifikowanego lub bankowości elektronicznej lub dokona uwierzytelnienia w punkcie potwierdzającym.

Profil zaufany (eGO) jest bezpłatną metodą uwierzytelniania obywateli w systemach e-administracji. Dzięki profilowi zaufanemu (eGO) klient urzędu może załatwić sprawę drogą elektroniczną bez konieczności posiadania płatnego odpowiednika, jakim jest podpis elektroniczny. Za jego pomocą można również zakładać konta w systemach zewnętrznych.

4. Przebieg badań

Celem badań jest ocena stopnia wykorzystania e-usług w administracji samorządowej w województwie lubelskim.

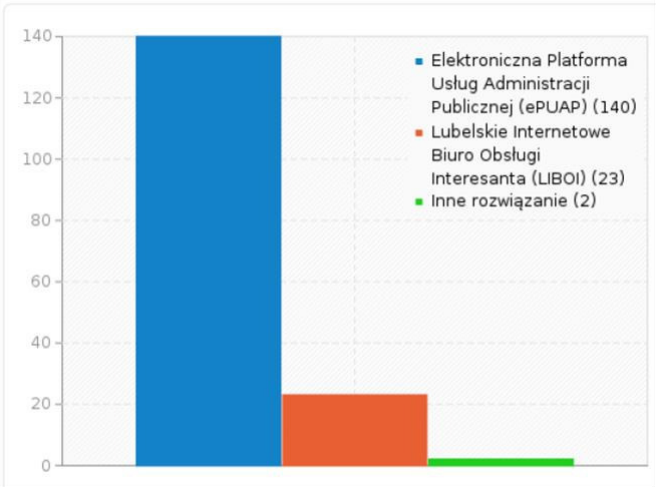
Badanie zostało przeprowadzone techniką CAWI (Computer Assisted Web Interview), której realizacja odbywa się za pomocą ankiety internetowej.

Ankietę przeprowadzono w okresie od marca do maja 2017 r. Link do ankiety internetowej przesłano 234 jednostkom samorządu terytorialnego województwa lubelskiego (JST WL). Ankietę internetową wypełniło 140 jednostek, co stanowi prawie 60% wszystkich urzędów w województwie.

5. Wyniki badań i ich analiza

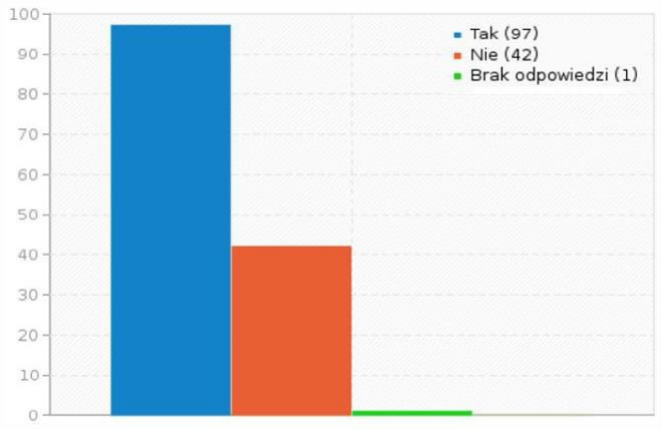
Analizując zebrane dane można stwierdzić, że JST WL posiadają bardzo dobrą podstawę do tworzenia, udostępniania i procedowania e-usług. Dowodem na to są wyniki ankiet, które wskazują, że wszystkie jednostki posiadają elektroniczną skrzynkę podawczą (ESP) w tym 100% jednostek, które przesłały ankietę wykorzystują ePUAP, 16% jednostek wykorzystuje również regionalne rozwiązanie

jakim jest LIBOI oraz 1% jednostek korzysta również z rozwiązań własnych (Rys. 2.).



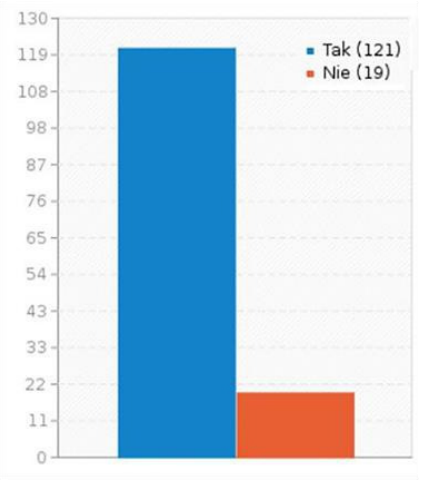
Rys. 2. Wykorzystanie elektronicznej skrzynki podawczej w JST WL

Respondenci zapytani zostali o intuicyjność w obsłudze wykorzystywanej przez nich ESP. 70% respondentów odpowiedziało, że wykorzystywana przez nich Elektroniczna Skrzynka Podawcza jest intuicyjna w obsłudze.



Rys. 3. Intuicyjność elektronicznych skrzynek podawczych

Jednostki informują swoich klientów o dostępności elektronicznej skrzynki podawczej w urzędzie. Na pytanie w ankiecie „Czy na stronie www urzędu w czytelny sposób prezentowany jest adres elektronicznej skrzynki podawczej?" 86% jednostek odpowiedziało twierdząco.



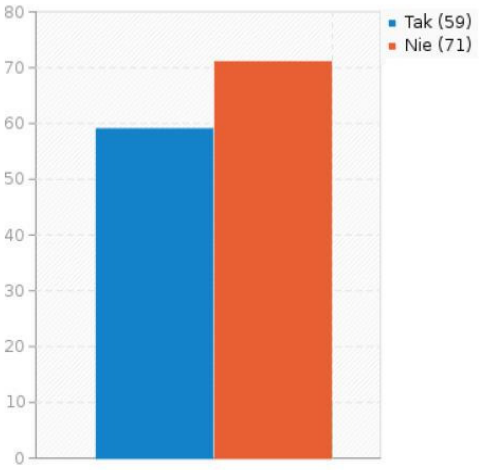
Rys. 4. Prezentacja w czytelny sposób na stronie www urzędu adresu elektronicznej skrzynki podawczej

Zadawalający jest również fakt posiadania przez JST WL podpisów elektronicznych, gdzie na 140 jednostek 139 wskazało liczbę posiadanych podpisów. Łączna liczby wszystkich podpisów posiadanych przez respondentów wynosi 1253.

Na pytanie w ankiecie, które dotyczyło pozytywnych skutków zastosowania Internetu w procesie świadczenia e-usług jednostki najczęściej wskazywały:

- skrócenie czasu realizacji usługi,
- wzrost liczby usług świadczonych drogą elektroniczną,
- obniżenie kosztów obsługi klienta,
- uproszczenie procedur obsługi klienta.

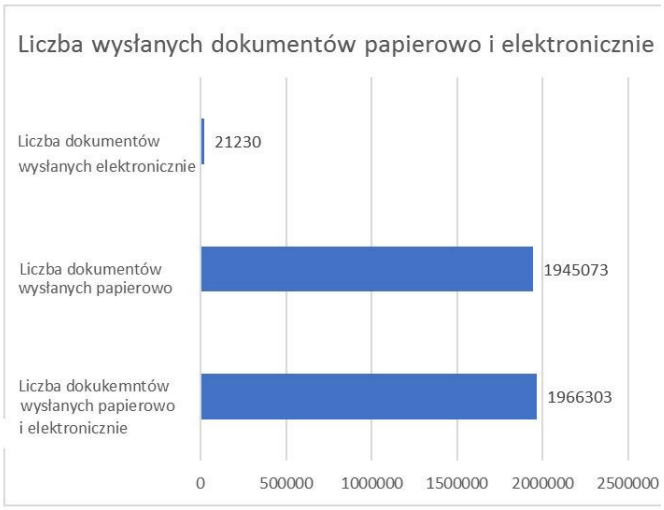
Na pytanie w ankiecie „Czy Państwa jednostka udostępniała obywatelom w 2016 r. usługi przez elektroniczną skrzynkę podawczą?" odpowiedziało 130 jednostek z czego 45% udostępniło w 2016 r. usługi na ESP, a 55% nie udostępniło żadnej usługi.



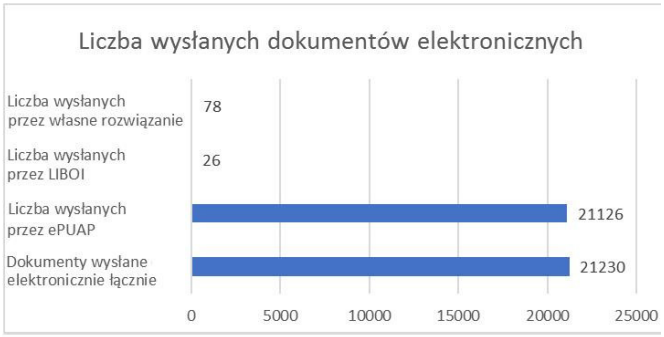
Rys. 5. Udostępnienie usług przez jedostki na elektronicznej skrzynce podawczej

Zapytano również o liczbę dokumentów wysłanych i odebranych zarówno tych papierowo jak i elektronicznie. Z poniższych danych wynika, że zaledwie 1% dokumentów

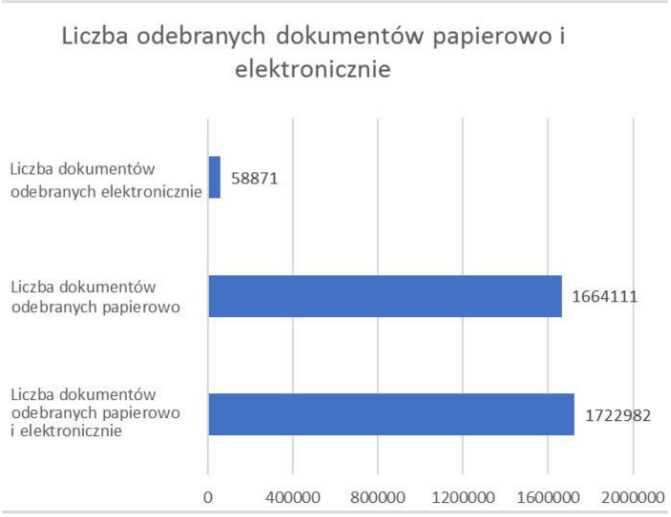
wysyłanych jest do obywatela elektronicznie. Niewiele lepszy jest wynik dotyczący dokumentów odebranych od obywatela elektronicznie bo wynosi on zaledwie 3%.



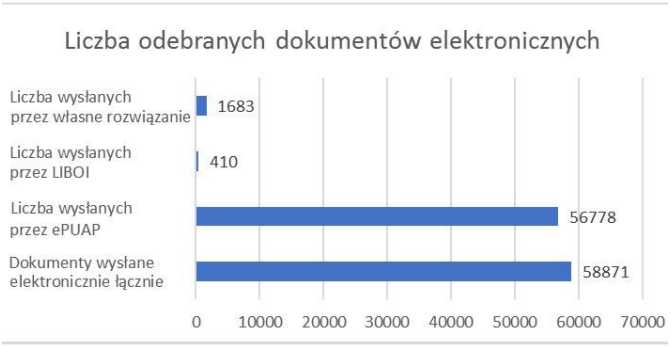
Rys. 6. Liczba wysłanych dokumentów papierowo i elektronicznie



Rys. 7. Liczba wysłanych dokumentów elektronicznych

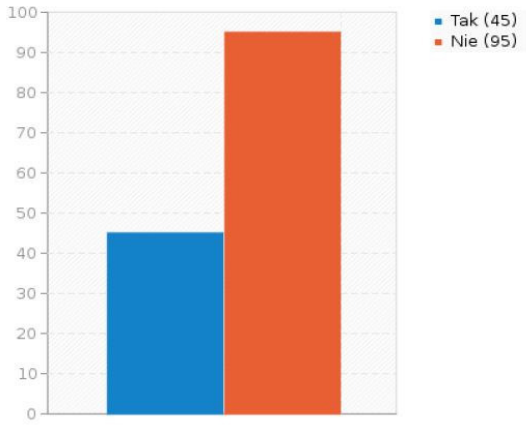


Rys. 8. Liczba odebranych dokumentów papierowo i elektronicznie



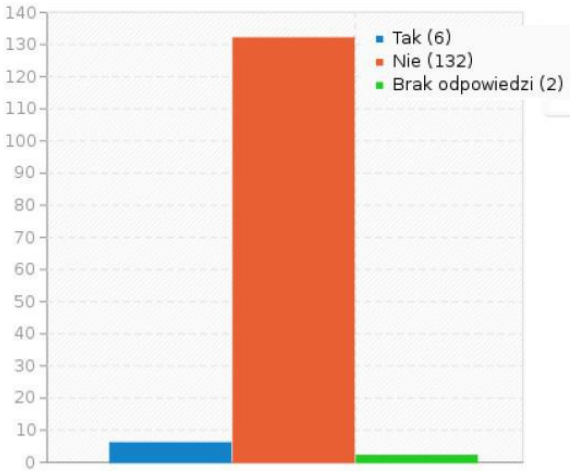
Rys. 9. Liczba odebranych dokumentów elektronicznych

Badania wskazują, że tylko 32% jednostek szkoli pracowników z zakresu wykorzystania i wdrażania usług elektronicznych. Niestety, aż 68% jednostek nie przeprowadza szkoleń dotyczących ww. zakresu.



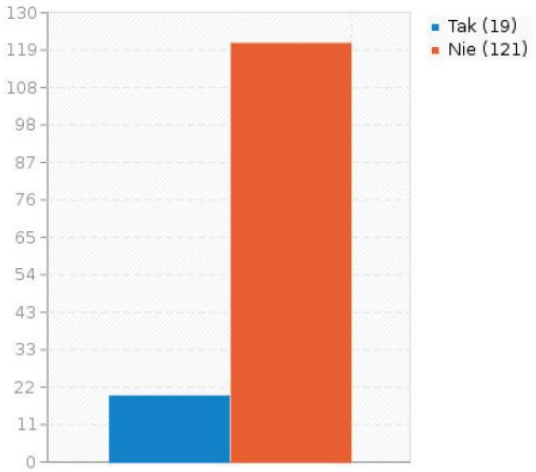
Rys. 10. Szkolenia obejmujące sposoby wykorzystania i wdrażania usług elektronicznych

Jak wynika z danych przeprowadzonych badań 96% jednostek nie ma wydzielonej osobnej komórki odpowiedzialnej za np. tworzenie, udostępnianie, nadzorowanie i procedowanie e-usług.



Rys. 11. Wydzielona komórka w jednostce odpowiedzialna za np. tworzenie, udostępnianie, nadzorowanie, procedowanie e-usług

Również znaczna części jednostek tj. 86 % nie posiada dokumentu strategicznego w zakresie informatyzacji.

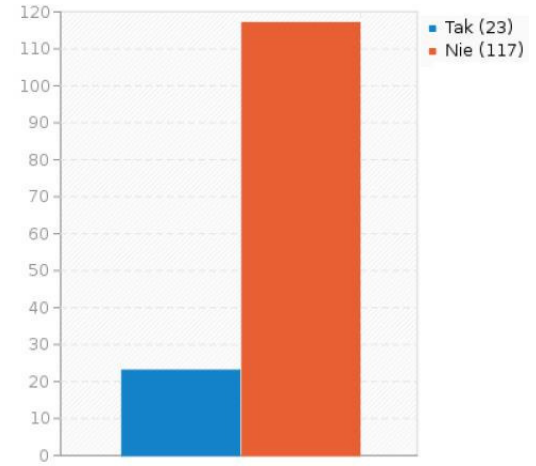


Rys. 12. Posiadanie przez JST WL dokumentu strategicznego w zakresie informatyzacji

W odpowiedzi na pytanie w ankiecie dotyczące problemów związanych z cyfryzacją, których doświadczył urząd w 2016 r. respondenci najczęściej zaznaczali:

- zbyt mały budżet na zadania związane z cyfryzacją działań i usług,
- wzrost wydatków na informatyzację w porównaniu z poprzednim analogicznym okresem,
- problem z działaniem systemów informatycznych,
- niewystarczającą liczbę osób do obsługi informatycznej w stosunku do potrzeb,
- zbyt skomplikowana/mało intuicyjna obsługa systemów informatycznych,
- niewystarczający poziom kompetencji informatycznych pracowników urzędu,
- problemy z wdrażaniem nowych systemów informatycznych.

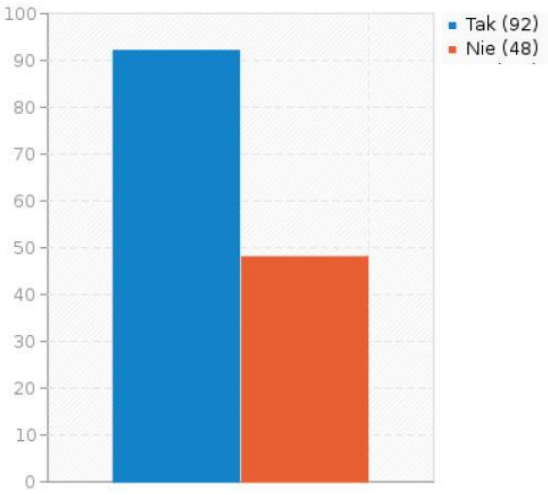
Wyniki ankiety wskazują, że 84 % JST WL nie tworzy własnych formularzy elektronicznych za pomocą narzędzi dostępnych w ESP.



Rys. 13. Tworzenie formularzy elektronicznych za pomocą narzędzi dostępnych na ESP

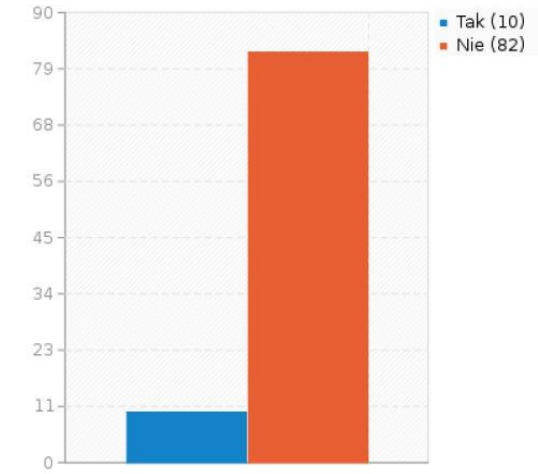
Najczęściej wskazywaną przyczyną, która jest barierą w tworzeniu formularzy jest brak procedur wewnętrznych związanych z tworzeniem, akceptacją i udostępnianiem formularzy elektronicznych. Na drugim miejscu jednostki wskazują zbyt skomplikowaną obsługę narzędzi do tworzenia formularzy elektronicznych, a ex aequo wskazywany jest brak wiedzy na temat dostępnych narzędzi do tworzenia formularzy elektronicznych oraz niewystarczającą liczbę osób do obsługi ESP.

Jedno z pytań ankiety brzmiało „Czy Państwa jednostka korzystała w 2016 r. z systemu elektronicznego zarządzania dokumentami?”. 66 % JST WL wskazało, że korzysta z systemu EZD.



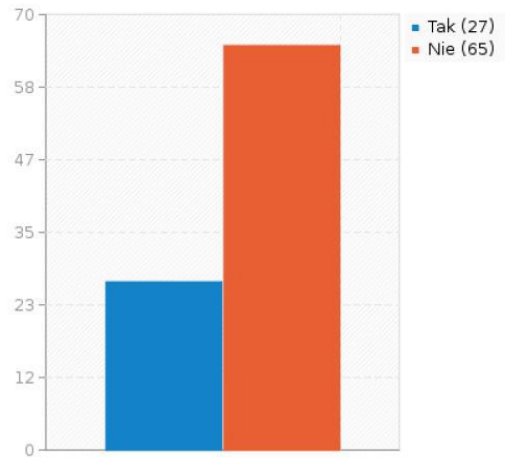
Rys. 14. Korzystanie z elektronicznego zarządzania dokumentami przez jednostki

Tylko 11 % jednostek korzysta z systemu elektronicznego zarządzania dokumentami, jako podstawowego sposobu dokumentowania przebiegu załatwiania i rozstrzygania spraw.



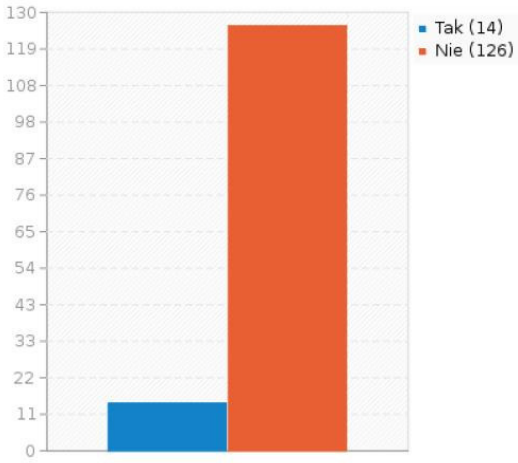
Rys. 15. Korzystanie z systemu EZD jako podstawowego sposobu dokumentowania przebiegu załatwiania i rozstrzygania spraw

Dodatkowo, aż 71 % jednostek, które odpowiedziały, że posiadają system EZD nie ma zintegrowanego systemu z elektroniczną skrzynką podawczą.



Rys. 16. Integracja systemu obiegu dokumentów z ESP

Niemała liczba, bo aż 90 % jednostek nie umożliwia weryfikacji statusu załatwianej sprawy.



Rys. 17. Weryfikacja statusu załatwianej sprawy

6. Wnioski

Mimo, iż wszystkie JST WL posiadają elektroniczną skrzynkę podawczą, gdzie większość respondentów potwierdziło jej intuicyjność oraz pozytywne zastosowanie umożliwiające między innymi skrócenie czasu realizacji usługi, obniżenie kosztów obsługi klienta, uproszenie procedur obsługi klienta to tylko 45 % jednostek udostępniło usługi na ESP. Zdaniem ankietowanych powodem takiego stanu rzeczy jest miedzy innymi zbyt mały budżet na zadania związane z cyfryzacją działań i usług w urzędzie, problem z działaniem systemów informatycznych, niewystarczająca liczba osób do obsługi informatycznej w stosunku do potrzeb czy też zbyt niski poziom kompetencji informatycznych pracowników urzędu.

Z przeprowadzanych ankiet wynika również, że JST WL w niskim stopniu dbają o podnoszenie kompetencji swoich pracowników nie zapewniając im odpowiednich szkoleń z zakresu wykorzystywania i wdrażania usług elektronicznych. Większość jednostek nie posiada dokumentu strategicznego w zakresie informatyzacji co również wpływa na brak odpowiednio obranej ścieżki rozwoju e-usług w urzędzie.

Dodatkowo tylko 16% JST WL tworzy i udostępnia formularze elektroniczne, które są wymagane w momencie świadczenia usług elektronicznych na poziomach 3 i 4. Zdaniem ankietowanych głównymi powodami braku tworzenia formularzy elektronicznych jest brak procedur wewnętrznych związanych z tworzeniem, akceptacją i udostępnianiem formularzy elektronicznych, zbyt skomplikowana obsługa narzędzi do tworzenia formularzy elektronicznych czy niewystarczająca liczba osób do obsługi tych narzędzi. Aby w pełni obsłużyć i zarządzać dokumentacją elektroniczną potrzebny jest system elektronicznego zarządzania dokumentami. Wyniki ankiet pokazały, że jest on wykorzystywany tylko przez 66% jednostek, z czego tylko 11% używa go jako systemu podstawowego. Pracownicy jednostek przez brak integracji elektronicznej skrzynki podawczej z systemem elektronicznego zarządzania dokumentami muszą obsługiwać dwa różne systemy, co jest mało komfortowym rozwiązaniem. Również brak integracji EZD z ESP (wykazane przez 90% jednostek) uniemożliwia sprawdzenie statusu załatwianej sprawy. Brak możliwości weryfikacji statusu może zniechęcić klientów do korzystania z e-usług oferowanych przez jednostki samorządu terytorialnego województwa lubelskiego.

Literatura

[1] Dz.U. 2016 poz. 1030. Obwieszczenie Marszałka Sejmu Rzeczypospolitej Polskiej z dnia 27 czerwca 2016 r. w sprawie ogłoszenia jednolitego tekstu ustawy o świadczeniu usług drogą elektroniczną.

[2] Rozporządzenie Rady (WE) nr 1777/2005 z dnia 17.10.2005 r. ustanawiające środki wykonawcze do dyrektywy 77/388/ EWG w sprawie wspólnego systemu podatku od wartości dodanej.

[3] https://epodrecznik.mac.gov.pl/mediawiki/index.php?title=E-us%C5%82uga [12.06.2017].

[4] http://eregion.wzp.pl/obszary/uslugi-elektroniczne [12.06.2017].

[5] Dz.U. 2005 nr 64 poz. 565, Ustawa z dnia 17 lutego 2005 r. o informatyzacji działalności podmiotów realizujących zadania publiczne.

[6] Janikowski A., Usługi ePUAP dla administracji publicznej, Wrocław 2011.

[7] http://wl.lubelskie.pl [15.06.2017].

[8] https://obywatel.gov.pl/o-serwisie [15.05.2017].