

Politechnika Poznańska
Wydział Informatyki
Instytut Informatyki

Praca dyplomowa inżynierska

**SYSTEM REALIZACJI ORAZ ANALIZY
TABLETOWEGO TESTU ŁĄCZENIA PUNKTÓW**

Paweł Białecki, 100017
Maciej Czarnecki, 99539
Maciej Jankowski, 99933
Paweł Olejniczak, 100351

Promotor
dr inż. Miłosz Kadziński

Poznań, 2014 r.

Tutaj przychodzi karta pracy dyplomowej.
Oryginał wstawiamy do wersji dla archiwum PP, w pozostałych kopiach wstawiamy ksero.

Spis treści

1	Wprowadzenie	1
1.1	Kontekst pracy	1
1.2	Cel i zakres pracy	2
1.3	Zespół i podział zadań	2
1.4	Struktura pracy	3
2	Sformułowanie problemu	5
2.1	Zastosowanie programu	5
2.2	Opis testu łączenia punktów	6
2.3	Zasada działania tabletu graficznego	7
3	Specyfikacja wymagań	9
3.1	Wymagania funkcjonalne	9
3.1.1	Opcje administracyjne	9
3.1.2	Realizacja badania	11
3.1.3	Obsługa danych	13
3.1.4	Analiza danych	15
3.2	Wymagania pozafunkcjonalne	15
4	Architektura aplikacji	17
4.1	Informacje ogólne	17
4.2	Przegląd zastosowanych technologii	17
4.2.1	Apache Maven	18
4.2.2	JavaFX	18
4.2.3	Hibernate	18
4.2.4	SQLite	19
4.2.5	JAXB	19
4.2.6	JPen	19
4.3	Szczegółowe omówienie struktury projektu	19
4.3.1	Model	20
4.3.2	Widok	21
4.3.3	Kontroler	22
5	Opis implementacji i realizacji projektu	25
5.1	Informacje ogólne i standard kodowania	25
5.2	Wybrane zagadnienia implementacyjne	26
5.2.1	Pobieranie danych z tabletu	26

5.2.2	Graficzny interfejs użytkownika	26
5.2.3	Kalibracja położenia kartki na tablecie	27
5.2.4	Odwzorowywanie toru kreślenia	28
5.2.5	Moduł statystyczny	29
5.2.6	Import i eksport danych	31
5.2.7	Metody przechowywania danych	31
5.3	Przebieg prac nad projektem	33
5.3.1	Informacje ogólne	33
5.3.2	Harmonogram prac i komunikacja w zespole	33
5.4	Zebrane doświadczenia	35
6	Użytkowa strona aplikacji	37
6.1	Główny ekran aplikacji	37
6.2	Logowanie użytkownika	38
6.3	Przeprowadzenie badania	39
6.4	Przegląd wyników badań	41
6.5	Moduł analizy danych	44
6.6	Import i eksport danych	47
6.6.1	Import	47
6.6.2	Eksport	47
6.7	Pozostałe funkcjonalności systemu	48
6.7.1	Dostęp do katalogów	48
6.7.2	Wylogowanie	48
7	Zakończenie	49
	Literatura	51
A	Załączniki	53

Rozdział 1

Wprowadzenie

1.1 Kontekst pracy

Kontrola ruchu i procesy nerwowe z nim związane poprzedziły w ewolucji inne procesy, które dziś nazywane są procesami psychicznymi. Z punktu widzenia informatyki, funkcje motoryczne, poznawcze i emocjonalne można interpretować jako procesy przetwarzania informacji. Podział na trzy rodzaje funkcji jest jednak często postrzegany jako sztuczny, ponieważ z poziomu układu nerwowego są one przetwarzane w podobny sposób. Co więcej, przy wyborze czynności motorycznej dochodzi do ich połączenia. Badanie zadań związanych z realizacją ruchu jest więc jednocześnie badaniem funkcji emocjonalnych i poznawczych. Opisując ruch, trzeba mieć świadomość, że zjawiska dotyczące czasu, przestrzeni oraz ilości są nierozzerwalne. Odpowiedzi na pytania jak długo, gdzie oraz ile pozwalają więc nie tylko na dobre scharakteryzowanie ruchu, ale również dostrzeżenie powiązań i wzajemnego przenikania się tych trzech wymiarów.

Zaburzenia psychomotoryczne polegają na niewłaściwej zależności między percepcją mózgową a działaniem. Ich podstawowych przyczyn można doszukiwać się w dwóch źródłach. Z jednej strony, istotnym czynnikiem jest tu stres, nadmierne tempo życia oraz eliminacja naturalnych elementów życia człowieka, co jest podłożem stanów nerwowych [Wol09]. Dodatkowo, niekorzystnym działaniem w takich sytuacjach jest nadmierne stosowanie leków lub innych używek, ponieważ może to w prosty sposób doprowadzić do uzależnień. Z drugiej strony, zaburzenia takie są konsekwencją chorób ośrodkowego układu nerwowego, jak np. choroba Alzheimera, polegająca na ogólnym otępieniu umysłu chorego, lub choroba Parkinsona, objawiająca się zaburzeniami równowagi i spowolnieniem procesów psychicznych.

W celu diagnozowania i kwantyfikacji zaburzeń psychomotorycznych powszechnie wykorzystuje się graficzne testy, z których najważniejszym jest test łączenia punktów (ang. *Trail Making Test*). Do tej pory realizowano go wyłącznie w formie papierowej, co znacznie utrudniało jego interpretację. Jedynym parametrem poddawanym analizie był całkowity czas trwania, przez co istniała możliwość tylko całościowej analizy przebiegu testu, bez dostępu do szczegółów badania np. czasu przebywania w danym punkcie lub wyznaczenia trudnych dla pacjenta przejść między punktami.

Przykład dotychczasowego pozyskiwania danych na podstawie testu łączenia punktów obrazuje, jak nieefektywne mogą być tradycyjne metody przeprowadzania testów medycznych. Biorąc pod uwagę rozwój techniki i dostęp do nowoczesnych narzędzi, jakość takich danych jest bardzo niska, a czas ich przetwarzania nieakceptowalnie długi. Dlatego też medycyna staje się jedną z głównych gałęzi zastosowań informatyki, dążąc do unowocześnienia metod pozyskiwania danych, ich analizy oraz wygodnego przechowywania. Łatwy dostęp do wyników badań pacjentów oraz ich

intuicyjna strukturalizacja mają bardzo duży wpływ na poziom świadczonych przez lekarzy usług medycznych.

Biorąc pod uwagę powyższe rozważania, powstała koncepcja stworzenia aplikacji umożliwiającej łatwe pozyskiwanie danych medycznych w postaci cyfrowej uzyskanych na podstawie testu łączenia punktów, ich szybką analizę i umożliwienie poprawnej interpretacji w celu postawienia trafnej diagnozy. Taki system powinien usprawnić pracę osób zajmujących się leczeniem pacjentów z zaburzeniami psychomotorycznymi, wpływającymi na niewłaściwą kooperację ośrodków ruchu z mózgiem.

1.2 Cel i zakres pracy

Celem przedsięwzięcia podjętego w ramach realizacji pracy inżynierskiej jest opracowanie systemu *TMT Analyzer* wspomagającego realizację badania polegającego na przeprowadzeniu testu łączenia punktów oraz rejestrację jego przebiegu za pomocą tabletu graficznego. Aplikacja dedykowana jest na komputery stacjonarne, posiadające dostęp do interfejsu USB, służącego do przeprowadzenia komunikacji z tabletem. Grupą docelową systemu są osoby zajmujące się leczeniem pacjentów z zaburzeniami psychomotorycznymi, wynikającymi np. z uzależnień, chorób cywilizacyjnych czy uwarunkowań genetycznych. Zadaniem aplikacji jest ponadto umożliwienie dokładnej analizy wartości statystycznych uzyskanych podczas realizacji testu przez pacjenta, wizualizacja toru jego kreślenia oraz wygodne przechowywanie danych medycznych. Użytkownicy dodatkowo uzyskają możliwość przeprowadzenia serii wielu badań i przeglądu ich wyników.

Najważniejsze wymagania funkcjonalne dotyczące aplikacji obejmują:

- umożliwienie komunikacji z tabletem graficznym, za pomocą którego system będzie miał możliwość rejestrowania przebiegu badania,
- zaprojektowanie intuicyjnego graficznego interfejsu użytkownika,
- opracowanie i implementacja modułu analizy danych statystycznych uzyskanych w badaniu,
- umożliwienie porównywania pary badań lub populacji pacjentów w oparciu o odpowiednie testy weryfikacji hipotez statystycznych,
- wizualizacja wyników badania oraz jego przebiegu,
- import i eksport danych przechowywanych w bazie.

1.3 Zespół i podział zadań

Projekt *TMT Analyzer* został zrealizowany przez zespół studentów Informatyki z Wydziału Informatyki Politechniki Poznańskiej w składzie:

- Paweł Bialecki,
- Maciej Czarnecki,
- Maciej Jankowski,
- Paweł Olejniczak.

Główne zadania, takie jak utworzenie graficznego interfejsu użytkownika, opracowanie funkcji wstępnego przetwarzania surowych danych czy kalibracja tabletu, realizowano w zespołach dwuosobowych. Pozostałe zadania wykonywano pojedynczo, z racji mniejszego nakładu pracy potrzebnego do ich realizacji. Omówmy teraz wkład i udział w pracy poszczególnych członków zespołu.

Paweł Białecki był odpowiedzialny za opracowanie graficznego interfejsu użytkownika, a więc zaprojektowanie części ekranów wyświetlanych przez system i zdefiniowanie przepływu sterowania, czyli zaimplementowanie kontrolerów do każdego ekranu. Posiadał również duży udział w tworzeniu procesu kalibracji położenia kartki z testem względem tabletu oraz wizualizacji toru kreślenia pacjenta. Ponadto, opracował i zaprogramował komunikację systemu z tabletem, a także zdefiniował typy strukturalne potrzebne do prawidłowej obsługi złożonych danych, np. szczegółowych wyników badania.

Maciej Czarnecki był kluczową postacią zespołu odpowiedzialną za organizację pracy podczas realizacji projektu. Zrealizował on dostęp do bazy danych z poziomu aplikacji i zaimplementował wizualizację przebiegu badania oraz import i eksport danych. Ponadto, korzystając z technologii *Apache Maven* oraz *XML*, opracował pliki odpowiedzialne za automatyczne zarządzanie projektem, jego testowanie i budowę. Był również odpowiedzialny za utworzenie klas wykorzystywanych podczas implementacji systemu, zdefiniowanych w warstwie modelu.

Maciej Jankowski był współodpowiedzialny za opracowanie graficznego interfejsu użytkownika. W tej kwestii, jego zadaniem było utworzenie dużej części ekranów wyświetlanych przez system i zdefiniowanie zależności między nimi, przy użyciu technologii *JavaFX*. Ponadto, jego główną odpowiedzialnością było zaimplementowanie funkcji przetwarzających surowe dane pochodzące z badania oraz utworzenie metod przeprowadzających określone testy statystyczne.

Paweł Olejniczak był współodpowiedzialny za implementację kalibracji położenia kartki z testem względem tabletu oraz wstępnego przetwarzania surowych danych. Dodatkowo, opracował on moduł szczegółowej analizy danych wraz z wyróżnieniem niezbędnych testów statystycznych. Był również odpowiedzialny za zaimplementowanie generowania szczegółowych raportów zawierających wyniki analizy danych w formacie PDF. Ponadto, jego zadaniem była również refaktoryzacja kodu programu.

1.4 Struktura pracy

Niniejsza praca inżynierska składa się z 7 rozdziałów, a jej struktura przedstawia się następująco:

Rozdział 2 jest poświęcony sformułowaniu problemu, którego rozwiązanie jest podstawą implementowanego systemu. Zawiera on szczegółowy opis testu łączenia punktów, jego zastosowania i znaczenie w medycynie. Rozważono tu także wpływ, jaki proponowane rozwiązanie może mieć na użytkowników programu oraz poziom świadczonych przez nich usług. Zawarto również opis działania tabletu graficznego, będącego kluczowym narzędziem podczas automatyzacji procesu przeprowadzania testu łączenia punktów.

Rozdział 3 zawiera szczegółową specyfikację wymagań funkcjonalnych wobec programu *TMT Analyzer*. Wyszczególniono w nim również opis wymagań нефункциональных, czyli warunków, które system powinien spełniać przy realizacji swoich funkcji, a także ograniczeń zewnętrznych.

Rozdział 4 przedstawia architekturę aplikacji. Zawarto w nim przegląd zastosowanych narzędzi i środowiska programowania. Wyróżniono w nim również opis poszczególnych warstw systemu,

zachowując podział na zdefiniowane pakiety klas.

Rozdział 5 poświęcono procesowi implementacji systemu *TMT Analyzer*. Omówiono w nim standard kodowania oraz scharakteryzowano wybrane, najciekawsze według zespołu, zagadnienia implementacyjne. Ponadto, zawarto w nim pełen opis przebiegu prac nad projektem, wraz z wyróżnieniem harmonogramu prac i schematu realizacji zadań. Uwzględniono tutaj napotkane problemy, a także kwestie dotyczące komunikacji między członkami zespołu oraz spostrzeżeń i doświadczeń nabytych podczas trwania prac nad projektem.

Rozdział 6 dotyczy użytkowej strony aplikacji, a więc zawiera dokładny opis całego przepływu sterowania w systemie oraz specyfikuje sposób, w jaki program pozwala na osiągnięcie wymaganej funkcjonalności.

W zakończeniu podsumowano osiągnięcie celów postawionych we wprowadzeniu do pracy, a także opisano sposób weryfikacji, czy zrealizowany system faktycznie stanowi rozwiązanie sformułowanego problemu.

Rozdział 2

Sformułowanie problemu

2.1 Zastosowanie programu

Projektowany system przeznaczony jest dla pracowników ośrodków zajmujących się leczeniem pacjentów z zaburzeniami psychomotorycznymi, wynikającymi z uzależnień, chorób cywilizacyjnych lub uwarunkowań genetycznych. Z założenia, podstawowym zastosowaniem programu jest automatyzacja przeprowadzania testu łączenia punktów oraz szczegółowa analiza wartości statystycznych uzyskanych na podstawie jego zarejestrowanego przebiegu. Obecnie, badanie pacjenta polega na realizacji testu w wersji papierowej, za pomocą długopisu. Taki sposób przeprowadzania badania powoduje możliwość jedynie całościowej jego analizy pod względem całkowitego czasu trwania. Wykorzystanie tabletu do realizacji testu łączenia punktów pozwala więc nie tylko na jego automatyzację, ale także uzyskanie szeregu dodatkowych parametrów opisujących psychomotorykę pacjenta oraz segmentację testu na mniejsze fragmenty. W przypadku tradycyjnej wersji papierowej, takie działanie nie byłoby możliwe, tak więc poprawna obsługa tabletu graficznego przez system stanowi fundamentalny element do przeprowadzenia procesu przetwarzania.

Ponadto, istotnym elementem aplikacji jest dedykowana baza danych przechowująca i utrzymująca w odpowiedniej strukturze wszystkie potrzebne informacje na temat pacjentów i przebiegu ich leczenia. Umożliwienie przez system szczegółowej analizy tych danych pozwala na ich właściwą interpretację, mającą na celu postawienie trafnej diagnozy medycznej, a więc wpływa bezpośrednio na poziom świadczonych przez użytkowników usług. Ważnym elementem systemu jest również możliwość przeprowadzenia importu lub eksportu opartego na zewnętrznych plikach przechowujących dane medyczne.

Kolejnym ważnym elementem jest wizualizacja toru kreślenia z wyróżnieniem popełnianych przez pacjenta błędów podczas realizacji testu. Funkcjonalność ta umożliwia użytkownikom jednoznaczne określenie problematycznych przejść (lub grup przejść) z punktu widzenia pacjenta oraz odtworzenie przebiegu badania w czasie przeglądania jego wyników.

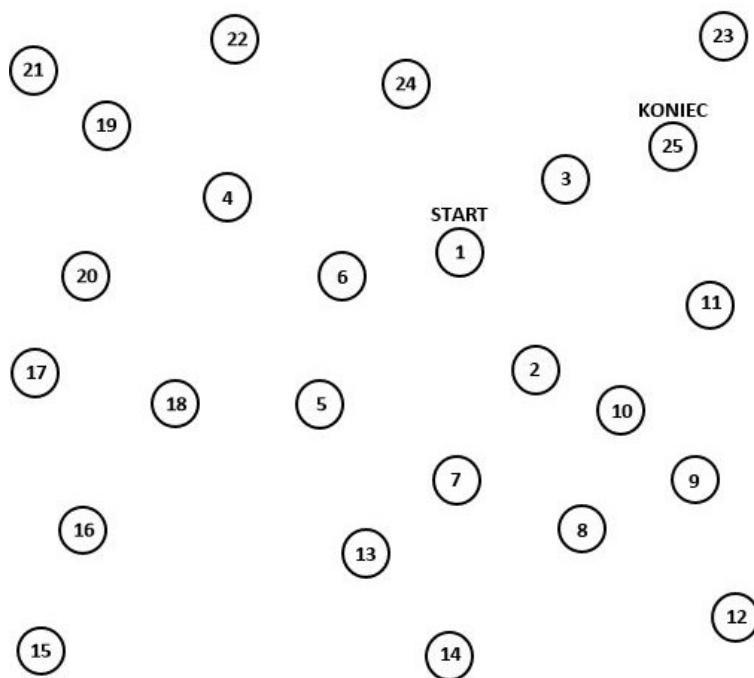
Istotnym elementem systemu jest dostęp do katalogów, pozwalających na sprawne definiowanie lub edytowanie danych w nich zawartych oraz ich filtracji mającej na celu zapewnienie użytkownikowi dostępu jedynie do danych stanowiących obiekt jego zainteresowania. Wyróżniono katalogi pacjentów, pracowników, chorób, leków oraz operacji. Dostęp do nich jest warunkowany rolą, jaką pełni dany użytkownik systemu.

2.2 Opis testu łączenia punktów

Test łączenia punktów - TMT (ang. *Trail Making Test*) to neuropsychologiczne badanie prędkości wzrokowego przeszukiwania i przetwarzania informacji, spostrzegawczości pacjenta oraz zdolności przełączania uwagi pomiędzy bodźcami różnego rodzaju. Test opracowano w 1944 roku jako część składową *Halstead-Retain Battery*, będącego zestawem testów badających zaburzenia neurologiczne. Dla potrzeb amerykańskiej armii zestaw ten poddano modyfikacji i nazwano *Army Individual Test Battery* [Mit05]. Obecnie są mu poddawani głównie pacjenci oddziałów neurologicznych, w tym jednostek zajmujących się leczeniem uzależnień oraz chorób z objawami zaburzeń psychomotorycznych. Dotychczas badanie polegało wyłącznie na zmierzeniu całkowitego czasu realizacji testu. Był on jedynym parametrem uzyskiwanym podczas testu i na którego podstawie wyciągano wnioski i stawiano diagnozy medyczne. Wyniki przeprowadzonych dotychczas badań potwierdzają, że wiek, edukacja oraz poziom inteligencji pacjentów mają istotny wpływ na całkowity czas realizacji testu [Tra14].

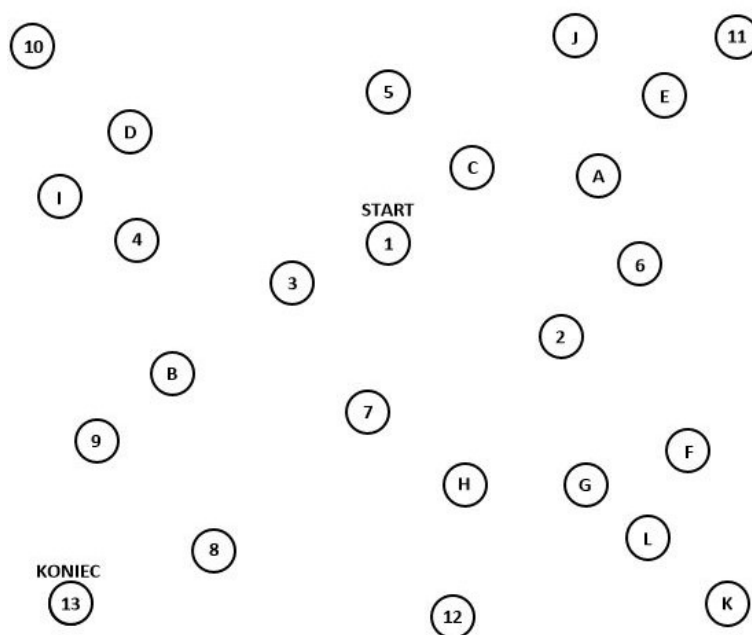
Test łączenia punktów składa się z dwóch części:

- część A - zadaniem pacjenta jest połączenie 25 punktów w kolejności numerycznej, tj. 1, 2, 3, ..., 25 (Rys. 2.1),
- część B - zadaniem pacjenta jest połączenie 25 punktów w kolejności numeryczno - literowej, tj. 1, A, 2, B, ..., 13 (Rys. 2.2).



Rysunek 2.1. Test łączenia punktów - część A

Część A bada przede wszystkim prędkość przetwarzania wzrokowego i sprawność koordynacji wzrokowo-ruchowej. Część B uznawana jest za nieco trudniejszą, ponieważ dodatkowo sprawdza



Rysunek 2.2. Test łączenia punktów - część B

reakcję na bodźce kilku rodzajów i umiejętność płynnego przełączania uwagi między nimi oraz występuje w niej dłuższa linia kreślenia.

Badanie pacjenta polega na przedstawieniu mu arkusza z wyrysowanymi punktami do połączenia wraz z instrukcjami odnośnie sposobu realizacji testu. Jego zadanie polega na jak najszybszej realizacji testu. W przypadku popełnienia błędu, jest on wskazywany przez osobę kontrolującą poprawność wykonania. Następnie, test jest kontynuowany od miejsca, do którego realizacja miała poprawny przebieg [Tra14]. Jeżeli osoba badana ma nadal problemy z właściwą realizacją testu, należy powtarzać procedurę tak długo, dopóki nie odniesie ona sukcesu lub dopóki nie okaże się, że dalsza realizacja testu jest niemożliwa do osiągnięcia.

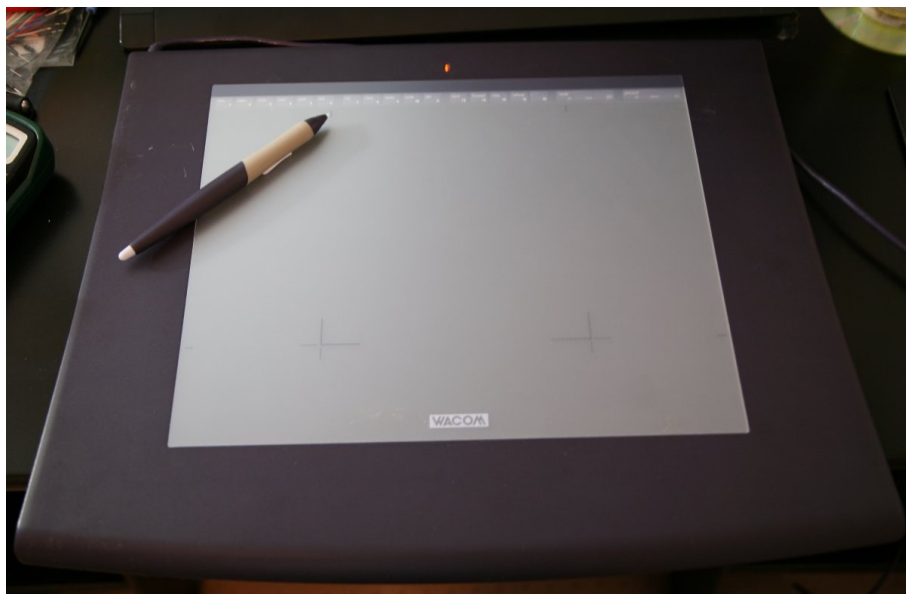
Warto podkreślić, że części A i B testu są realizowane przez danego pacjenta bezpośrednio po sobie. Następnie, po podaniu przez lekarza środków medycznych albo wykonaniu operacji, badanie realizowane jest ponownie. Ma to na celu sprawdzenie skutków działania zastosowanej terapii lekowej lub zabiegu. Badaniu poddawani są pacjenci zgrupowani w serie zależne od daty realizacji badania lub jego lokalizacji. Celem takiego działania jest poprawna organizacja danych uzyskanych z testu.

2.3 Zasada działania tabletu graficznego

Tablet graficzny to urządzenie służące do precyzyjnego ustalania pozycji kursora na ekranie. Jego podstawowym zastosowaniem jest artystyczna grafika komputerowa polegająca np. na przetwarzaniu i modyfikacji obrazów. Tablet na swojej powierzchni posiada kilka wyrysowanych znaczników, umożliwiających precyzyjne określenie zakresu manewrowania kursorem lub umieszczenie kartki formatu A4 we właściwy sposób. Do tabletu dołączony jest zestaw piór wytwarzających

niewielkie pole magnetyczne mające na celu ich poprawną lokalizację na powierzchni urządzenia. Niektóre pióra zawierają dodatkowo standardowy atrament. W taki sposób można się nimi posługiwać jak zwykłymi długopisami.

Tabletem wykorzystywanym do testów realizowanego systemu jest *Intuos2* firmy *Wacom*. W przypadku tego urządzenia, zakres manewrowanie kursorem jest uzależniony od rozdzielczości ekranu komputera, do którego je podłączono. Tablet graficzny standardowo do komunikacji z komputerem wykorzystuje interfejs USB. Nie wymaga on instalowania dodatkowego oprogramowania lub sterowników, umożliwiających normalną pracę. Na Rysunku 2.3 przedstawiono opisywany tablet.



Rysunek 2.3. Tablet graficzny wykorzystywany do testów systemu

Dane rejestrowane przy wykorzystaniu tabletu mogą być wykorzystane przez programy i posiadają strukturę pakietu *htd*, którego budowę przedstawiono w tabeli 2.1.

Tablica 2.1
Struktura zawierająca dane pochodzące z tabletu

Pole	Opis
int pkTime	Dokładny moment pobrania danych z tabletu, przyjmuje wartości zależne od zegara systemowego komputera.
int pkX	Współrzędna X pióra na tablecie
int pkY	Współrzędna Y pióra na tablecie <i>Współrzędne X i Y oraz ich zasięgi są uzależnione od rozdzielczości ekranu komputera, do którego podłączono tablet.</i>
int pkPressure	Stopień nacisku pióra na tablet, podlega normalizacji i dlatego przyjmuje wartości z zakresu $[0,1]$, gdzie 1 to maksymalny nacisk obsługiwany przez tablet.
int pkAzimuth	Kąt zawarty pomiędzy osią X a projekcją pióra na płaszczyźnie X-Y. Posiada kierunek zgodny z ruchem wskazówek zegara. Zakres: $[-\pi/2, 3/2 * \pi]$.
int pkAltitude	Kąt zawarty pomiędzy osią pióra a jego projekcją na płaszczyźnie X-Y. Zakres: $[0, \pi/2]$.

Rozdział 3

Specyfikacja wymagań

3.1 Wymagania funkcjonalne

Wymagania funkcjonalne (FRs) opisują operacje realizowane przez system [Jas97]. Zawierają pełen opis funkcjonalności systemu widzianej od strony użytkownika oraz posiadają określone zależności czasowe i kryteria akceptacji. Metoda ich grupowania polega na zdefiniowaniu przynależności danej funkcji programu do głównych kategorii zadań. W systemie wyróżniono kilka takich kategorii, mających na celu odpowiednią strukturalizację wymagań.

3.1.1 Opcje administracyjne

Dostęp do danych oraz funkcjonalności udostępnianych przez *TMT Analyzer* jest ograniczony koniecznością zalogowania. Weryfikacja tożsamości użytkownika polega na podaniu jego nazwy i przypisanego mu hasła. W zależności od zdefiniowanej w systemie roli, użytkownik posiada dostęp jedynie do określonego zakresu funkcjonalności.

Tablica 3.1
FR1

Nazwa	Logowanie użytkownika.
Opis	System powinien zapewnić obsługę logowania użytkownika za pomocą niepowtarzalnej nazwy oraz hasła.
Uzasadnienie	Logowanie użytkownika ma na celu zapobieganie niepowołanego dostępu do wrażliwych danych medycznych oraz weryfikację zakresu funkcjonalności udostępnianego dla danej osoby w zależności od jej roli w systemie.
Kryterium akceptacji	Tuż po uruchomieniu systemu wyświetlane jest okno logowania, w którym użytkownik powinien podać swoją nazwę i hasło.
Priorytet ważności	Powinno zostać wdrożone
Zależności	Brak
Konflikty	Brak

Tablica 3.2
FR2

Nazwa	Podział użytkowników na grupy.
Opis	Aplikacja powinna definiować role administratora systemu oraz użytkownika - lekarza.
Uzasadnienie	Zadaniem podziału na grupy jest określenie zakresów odpowiedzialności użytkowników oraz dostępnych dla nich funkcjonalności.
Kryterium akceptacji	Użytkownik, w zależności od typu konta na jakie się zaloguje, powinien mieć zróżnicowany dostęp do funkcjonalności systemu.
Priorytet ważności	Może zostać wdrożone
Zależności	FR1
Konflikty	Brak

Tablica 3.3
FR3

Nazwa	Dostęp do katalogu pracowników z poziomu administratora.
Opis	Użytkownik posiadający rolę administratora systemu powinien mieć dostęp do listy wszystkich zdefiniowanych użytkowników w systemie.
Uzasadnienie	Administrator powinien posiadać możliwość definiowania nowych oraz edycji i usuwania istniejących użytkowników w celu zapewnienia pracownikom prawidłowego dostępu do systemu.
Kryterium akceptacji	Użytkownik zalogowany jako administrator, po wybraniu przycisku Opcji może uzyskać dostęp do katalogu pracowników.
Priorytet ważności	Może zostać wdrożone
Zależności	FR2
Konflikty	Brak

3.1.2 Realizacja badania

Automatyzacja procesu przeprowadzania badania to jedno z podstawowych zadań systemu *TMT Analyzer*. Jej głównym zadaniem jest rejestracja przebiegu testu, umożliwiającą przeprowadzenie dalszej analizy zgromadzonych danych. Dodatkowo, wsparcie użytkowników podczas przeprowadzania badań ma na celu szybsze wychwytywanie błędów i wygodniejsze przeprowadzanie serii takich testów.

Tablica 3.4
FR4

Nazwa	Kalibracja pozycji kartki względem tabletu.
Opis	Aplikacja powinna umożliwiać kalibrację tabletu polegającą na sprawdzeniu względnego położenia arkusza z testem.
Uzasadnienie	Osoba przeprowadzająca badanie musi mieć pewność, że uzyskane wyniki są prawidłowe i mają odpowiedni punkt odniesienia w przypadku np. niewłaściwego montażu kartki na tablecie.
Kryterium akceptacji	Użytkownik dokonuje kalibracji zgodnie z wyświetlanymi na ekranie instrukcjami przed realizacją każdego testu oraz w razie swojej potrzeby.
Priorytet ważności	Musi zostać wdrożone
Zależności	Brak
Konflikty	Brak

Tablica 3.5
FR5

Nazwa	Przeprowadzanie badania.
Opis	Użytkownik musi mieć możliwość rejestracji przebiegu badania oraz zdefiniowania niezbędnych parametrów do jego prawidłowego przeprowadzenia, np. wyboru pacjenta lub określenia serii badań.
Uzasadnienie	Rejestracja przebiegu badania umożliwia aplikacji odwzorowanie toru kreślenia, szczegółową analizę danych oraz ich wizualizację. Dodatkowo, pozwala na określenie podstawowych danych dotyczących badania, na które składają się: dane personalne pacjenta, seria badań czy zastosowane środki medyczne.
Kryterium akceptacji	Użytkownik po wciśnięciu odpowiedniego przycisku najpierw dokonuje kalibracji tabletu, następnie wybiera serię badań i określa dane pacjenta, po czym posiada dostęp do okna z wyrysowanymi punktami i linią kreślenia pacjenta.
Priorytet ważności	Musi zostać wdrożone
Zależności	FR4
Konflikty	Brak

Tablica 3.6
FR6

Nazwa	Odwzorowywanie toru kreślenia pacjenta.
Opis	System powinien umożliwiać rysowanie toru kreślenia pacjenta w trybie on-line na podstawie pakietów zawierających dane o położeniu pióra.
Uzasadnienie	Odwzorowywanie toru ruchu pióra może być bardzo pomocne dla użytkownika do określenia szczególnie problematycznych przejść między punktami oraz może stanowić podstawę późniejszej szczegółowej analizy danych.
Kryterium akceptacji	Podczas rejestracji przebiegu badania użytkownik posiada dostęp do okna z umieszczonymi punktami z testu. Wówczas, wraz z ruchem ręki pacjenta następuje rysowanie toru kreślenia pióra po tablecie.
Priorytet ważności	Powinno zostać wdrożone
Zależności	FR5
Konflikty	Brak

Tablica 3.7
FR7

Nazwa	Obsługa zakończenia badania.
Opis	System po zakończeniu badania powinien umożliwiać ponowną jego realizację oraz zaakceptowanie lub odrzucenie jego przebiegu.
Uzasadnienie	Po zakończeniu badania użytkownik powinien mieć możliwość odrzucenia jego wyników lub ponownej realizacji w przypadku dużej liczby błędów pacjenta lub oddziaływania na niego niekorzystnych czynników zewnętrznych. W przeciwnym razie, zarejestrowany przebieg powinien zostać zapisany do bazy danych.
Kryterium akceptacji	Po zakończeniu badania system wyświetla okno z trzema opcjami do wyboru: odrzucenie przebiegu testu, jego akceptacja lub chęć ponownej realizacji.
Priorytet ważności	Powinno zostać wdrożone
Zależności	FR5
Konflikty	Brak

3.1.3 Obsługa danych

Dostęp do danych zgromadzonych podczas badań oraz ich sprawna obsługa mają za zadanie ułatwienie pracy użytkowników i wspieranie ich w trakcie stawiania diagnoz medycznych, co ma bezpośredni wpływ na poziom świadczonych przez nich usług. Dostęp do wyników badań pozwala na monitorowanie i weryfikację postępów pacjenta zachodzących wskutek zastosowanej kuracji.

Tablica 3.8
FR8

Nazwa	Moduł przeglądu badań.
Opis	System powinien umożliwiać przejrzanie szczegółowych wyników badań pacjentów.
Uzasadnienie	Dostęp do szczegółowych wyników badań pacjentów zawartych w bazie danych umożliwia podjęcie prawidłowej decyzji odnośnie zastosowania środków medycznych mających na celu poprawę stanu pacjenta.
Kryterium akceptacji	Po wciśnięciu odpowiedniego przycisku użytkownik wybiera interesującego go pacjenta. Wówczas system wyświetla nowe okno zawierające szczegółowe dane dotyczące przebiegu badania wraz z ich wizualizacją i narysowanym torem kreślenia pacjenta.
Priorytet ważności	Musi zostać wdrożone
Zależności	Brak
Konflikty	Brak

Tablica 3.9
FR9

Nazwa	Filtracja badań.
Opis	Aplikacja powinna umożliwiać nałożenie filtrów na katalog badań i wyświetlanie wyłącznie tych rekordów, które są interesujące dla użytkownika. Filtrami mogą być np. lekarz prowadzący, lokalizacja badania lub data.
Uzasadnienie	Istnieje wysokie prawdopodobieństwo przechowywania dużej liczby badań oraz pacjentów, w związku z czym potrzebny jest mechanizm umożliwiający przefiltrowanie wyników i wyświetlenie jedynie tych, które mogą być potrzebne użytkownikowi.
Kryterium akceptacji	Użytkownik po przejściu do modułu przeglądu badań może, poniżej tabeli z wyszczególnionymi pacjentami, wybrać jedną z udostępnianych przez system metod filtracji.
Priorytet ważności	Powinno zostać wdrożone
Zależności	FR8
Konflikty	Brak

Tablica 3.10
FR10

Nazwa	Import i eksport danych.
Opis	Aplikacja powinna udostępniać możliwość importu danych do bazy z zewnętrznego pliku tekstowego oraz, ponadto, eksportu danych do takiego pliku.
Uzasadnienie	Potrzeba wdrożenia importu danych do bazy wynika z faktu częstego przechowywania wyników badań w plikach tekstowych o strukturze <i>htd</i> , natomiast eksport danych z bazy może być przydatny podczas przenoszenia danych na inny komputer.
Kryterium akceptacji	Użytkownik wybiera interesującą go funkcję z głównego okna aplikacji, po czym wybiera plik do importu lub zakres danych, które chce z bazy wyeksportować.
Priorytet ważności	Musi zostać wdrożone
Zależności	Brak
Konflikty	Brak

Tablica 3.11
FR11

Nazwa	Dostęp do katalogów.
Opis	Aplikacja powinna umożliwiać dostęp do katalogów pacjentów, leków, chorób i operacji.
Uzasadnienie	Zastosowanie katalogów powinno ułatwić użytkownikom korzystanie z aplikacji, bez potrzeby ciągłego wpisywania tych samych wartości oraz zapewniając łatwe utrzymanie struktury danych przydatnych podczas leczenia pacjenta.
Kryterium akceptacji	W głównym oknie aplikacji użytkownik wybierając przycisk Opcji uzyskuje dostęp do listy rozwijanej, z której może wybrać interesujący go katalog.
Priorytet ważności	Powinno zostać wdrożone.
Zależności	Brak
Konflikty	Brak

3.1.4 Analiza danych

Szczegółowa analiza zgromadzonych danych jest najważniejszą częścią systemu z punktu widzenia jego logiki działania. Pozwala na dokładne sprawdzenie postępów pacjenta i różnic w badaniach nad nim przeprowadzonych. Ponadto, udostępnienie modułu analizy danych stanowi podstawę do weryfikacji podobieństw między wybranymi przez użytkownika populacjami.

Tablica 3.12
FR12

Nazwa	Możliwość parowania populacji.
Opis	System powinien umożliwiać funkcjonalność parowania populacji w celu poprawnego przeprowadzenia testów statystycznych.
Uzasadnienie	Parowanie pacjentów jest niezbędnym elementem do przeprowadzenia testów statystycznych badających różnice między populacjami oraz między wynikami dwóch przebadanych osób.
Kryterium akceptacji	Użytkownik po uruchomieniu modułu analizy danych zaznacza pacjentów których chce poddać testowi, po czym wybiera przycisk 'Dodaj' znajdujący się poniżej. Wówczas, w dedykowanej do tego tabeli, pojawia się określona przez użytkownika para pacjentów.
Priorytet ważności	Musi zostać wdrożone
Zależności	Brak
Konflikty	Brak

Tablica 3.13
FR13

Nazwa	Przeprowadzenie analizy danych.
Opis	Aplikacja powinna realizować testy weryfikacji hipotez statystycznych na danych określonych przez użytkownika podczas procesu parowania populacji.
Uzasadnienie	Potrzeba wdrożenia modułu analizy danych wynika z założeń projektowanego systemu. Weryfikacja różnic pomiędzy określonymi badaniami lub populacjami to jedna z głównych funkcjonalności, które powinien on udostępniać.
Kryterium akceptacji	Po ukończeniu procesu parowania populacji ekran wyświetla pasek postępu realizacji testu, po czym wyświetlany jest raport z jego przebiegu. Zawiera on decyzję o odrzuceniu hipotezy statystycznej, a także informacje dotyczące poziomu istotności, rodzaju zastosowanego testu oraz jego p-wartości.
Priorytet ważności	Musi zostać wdrożone
Zależności	FR12
Konflikty	Brak

3.2 Wymagania pozafunkcjonalne

Wymagania pozafunkcjonalne (NFRs) opisują ograniczenia, przy których system ma realizować swoje funkcje, a osoby opracowujące program muszą uwzględnić w czasie pracy nad projektem [Jas97]. Wymagania pozafunkcjonalne ponadto określają warunki efektywnego wykorzystania systemu, a także definiują zalecenia dotyczące bezpieczeństwa aplikacji.

Tablica 3.14
NFR1

Nazwa	Harmonogramowanie projektu.
Opis	Rozpoczęcie implementacji systemu z dniem 1 września 2013 roku oraz ustalenie terminu ukończenia wszystkich prac na 1 lutego 2014 roku.
Priorytet ważności	Musi zostać wdrożone

Tablica 3.15
NFR2

Nazwa	Zapewnienie bezproblemowej instalacji systemu.
Opis	Weryfikacja, czy proces instalacji systemu odbywa się w sposób sprawny, niewymagający dodatkowych sterowników ani oprogramowania odpowiedzialnego za jego poprawną pracę.
Priorytet ważności	Musi zostać wdrożone

Tablica 3.16
NFR3

Nazwa	Zapewnienie przenoszalności systemu.
Opis	Umożliwienie korzystania z aplikacji niezależnie od ustawionej rozdzielczości ekranu lub mocy obliczeniowej komputera.
Priorytet ważności	Powinno zostać wdrożone

Tablica 3.17
NFR4

Nazwa	Intuicyjny graficzny interfejs użytkownika.
Opis	Interfejs aplikacji powinien być funkcjonalny oraz czytelny.
Priorytet ważności	Powinno zostać wdrożone

Tablica 3.18
NFR5

Nazwa	Opracowanie formatu danych.
Opis	Stworzenie jednolitego formatu ułatwiającego import oraz eksport danych przechowywanych w bazie.
Priorytet ważności	Powinno zostać wdrożone

Rozdział 4

Architektura aplikacji

4.1 Informacje ogólne

Realizując aplikację, zdecydowano się na zastosowanie architektonicznego wzorca projektowego *Model-View-Controller* (MVC). Jest to wzorzec efektywnie łączący interfejs użytkownika, czyli warstwę prezentacji z zasadniczym modelem danych. Polega on na podziale komponentów systemu na trzy kategorie: (1) komponenty typu Model reprezentują dane logiki biznesowej, na której operuje aplikacja, (2) komponenty typu Widok są odpowiedzialne za prezentację danych dla użytkownika, (3) komponenty typu Kontroler to komponenty przechytujące żądania użytkowników i odwzorowujące je w wywołania metod komponentów typu Model. Następnie, komponenty typu Kontroler przekazują sterowanie do komponentów typu Widok, umożliwiając wizualizację danych, które interesują użytkownika [Zak06]. Komponenty typu Kontroler odpowiadają również za realizację funkcjonalności projektu. Wzorzec projektowy MVC jest często stosowany przez wielu profesjonalnych programistów z racji dobrej organizacji kodu oraz znaczącej redukcji czasu potrzebnego do tworzenia aplikacji posiadających interfejs użytkownika. Odpowiednia separacja warstw aplikacji decyduje o niezależnym tworzeniu poszczególnych sekcji kodu, co ma bezpośredni wpływ na jego jakość. Podsumowując, na zastosowanie wzorca *Model-View-Controller* wpływ miały poniższe czynniki:

- znajomość wzorca z poprzednio realizowanych projektów,
- lepsza organizacja kodu,
- separacja warstw aplikacji,
- łatwość w rozwijaniu i ponownym użyciu kodu,
- łatwiejsze testowanie.

4.2 Przegląd zastosowanych technologii

W tym rozdziale przedstawiono krótką charakterystykę najważniejszych technologii i narzędzi zastosowanych podczas tworzenia systemu *TMT Analyzer*.

4.2.1 Apache Maven

Apache Maven - narzędzie automatyzujące budowę oprogramowania na platformę Java. Odpowiada on za dwa aspekty tworzenia oprogramowania: określa jak jest ono tworzone oraz opisuje jego zależności. Plik określający sposób budowy aplikacji, jej zależności oraz zewnętrzne moduły i komponenty nosi nazwę (ang. *Project Object Model*) i definiowany jest w formacie XML (przykład zawarto na Listingu 4.1). *Maven* pobiera biblioteki i wtyczki w sposób dynamiczny z centralnych repozytoriów i przechowuje je w lokalnym cache-u. Jego zastosowanie podczas realizacji systemu wspomagało korzystanie z dobrych praktyk tworzenia projektu, domyślnie wymagając określonej struktury katalogów oraz ułatwiało proces organizacji i budowania projektu.

```
1 <project>
2   <modelVersion>4.0.0</modelVersion>
3   <groupId>com.mycompany.app</groupId>
4   <artifactId>my-app</artifactId>
5   <version>1</version>
6 </project>
```

LISTING 4.1: Przykładowy plik pom.xml.

4.2.2 JavaFX

JavaFX - technologia powstała z myślą o tworzeniu bogatych aplikacji internetowych, dedykowanych dla różnych platform i urządzeń. Dostarcza ona język programowania wysokiego poziomu wraz z interfejsem programowania aplikacji. Jest to specyfikacja Javy pochodząca z firmy Oracle, będąca bezpośrednią konkurencją Adobe Flash i Microsoft Silverlight, również przeznaczonych do tworzenia bogatych aplikacji internetowych. *JavaFX* może być używana na różnych urządzeniach i jest realizowana jako część środowiska wykonawczego Java w niektórych urządzeniach mobilnych. Podczas realizacji systemu zastosowano tę technologię, w celu stworzenia graficznego interfejsu użytkownika. Do projektowania poszczególnych okien skorzystano z narzędzia dostarczanego przez technologię - *JavaFX Scene Builder*.

4.2.3 Hibernate

Hibernate - platforma programistyczna realizująca warstwę dostępu do danych. Zapewnia możliwość translacji danych pomiędzy relacyjną bazą danych a obiektowymi językami programowania. Opiera się na wykorzystaniu opisu struktury danych za pomocą języka *XML* oraz adnotacji w kodzie *Javy*, dzięki czemu można rzutować obiekty bezpośrednio na istniejące tabele danych [Hib14]. Pozwala realizować w szybki sposób obsługę przechowywania danych w bazie, niezależnie od systemu baz danych. Dodatkowo, *Hibernate* zwiększa wydajność operacji na bazie danych dzięki buforowaniu i minimalizacji liczby przesyłanych zapytań. Podczas implementacji projektu, zastosowano tę technologię do realizacji mapowania obiektowo-relacyjnego.

4.2.4 SQLite

SQLite - system zarządzania bazą danych implementujący większość standardów SQL. Jest to jeden z najpopularniejszych systemów wykorzystywanych w oprogramowaniu. SQLite nie posiada odrębnego procesu z którym aplikacja może się komunikować, ale staje się jej integralną częścią, przez co optymalizowany jest dostęp do danych. Podczas modyfikacji zawartości bazy danych, następuje jej całkowita blokada, mająca na celu zapewnienie spójności danych. W trakcie realizacji projektu zastosowano system SQLite do utworzenia i zarządzania bazą danych ze względu na brak konieczności jego instalacji i niewielkie zapotrzebowanie na zasoby.

4.2.5 JAXB

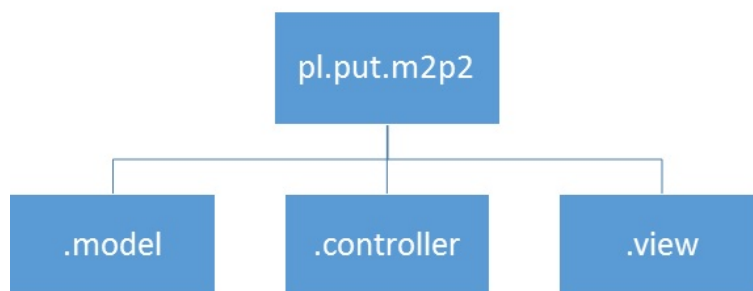
JAXB (ang. *Java Architecture for XML Binding*) - specyfikacja konwertowania obiektów w języku Java na ich odpowiedniki w formacie *XML* oraz zamiany obiektów w formacie *XML* na obiekty w języku Java. Pozwala na wykonywanie powyższych operacji bez konieczności głębszej znajomości sposobu przetwarzania plików w formacie *XML*. JAXB pozwala na modyfikacje przebiegu konwersji między reprezentacjami danych. Podczas realizacji systemu, skorzystano z tej technologii podczas implementacji modułu importu i eksportu danych.

4.2.6 JPen

JPen - biblioteka umożliwiająca dostęp z poziomu aplikacji do tabletów graficznych i urządzeń wskazujących przy użyciu technologii Java. Opiera się ona na architekturze *Event/Listener* polegającej na realizacji działania po nadejściu pewnego zdarzenia zarejestrowanego przez obiekt nasłuchujący. Dostęp do urządzeń wskazujących jest zaimplementowany przez sterowniki. JPen zawiera sterowniki dla systemów operacyjnych Linux, Windows i Max OS X.

4.3 Szczegółowe omówienie struktury projektu

Zaprojektowanie całej struktury wewnętrznej systemu należało do zadań zespołu programistycznego. Architekturę podzielono na trzy odseparowane od siebie logicznie i współpracujące ze sobą warstwy Modelu, Widoku i Kontrolera (Rys. 4.1).



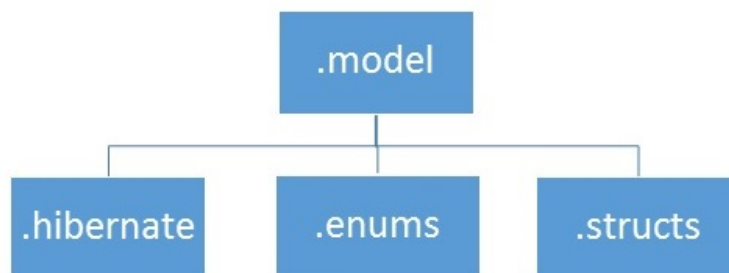
Rysunek 4.1. Podział architektury systemu na trzy warstwy.

W warstwie Modelu i Kontrolera zdefiniowano pakiety klas *Javy*, posiadające określony zakres odpowiedzialności i zawierające definicje konkretnych funkcjonalności. W przypadku warstwy Widoku nie zdecydowano się na utworzenie pakietów, ponieważ nie zawarto w niej żadnej klasy *Javy*. Znajdują się tutaj definicje poszczególnych okien zrealizowane przy pomocy plików *.xml*, style *.css* odpowiedzialne za ustalenie odpowiedniego stylu interfejsu graficznego oraz obrazy wyświetlane przez system.

W kolejnych podrozdziałach przedstawiono szczegółowe informacje dotyczące poszczególnych warstw systemu *TMT Analyzer*. Warstwy opisano według podziału na pakiety (przestrzenie nazw).

4.3.1 Model

Pierwszą warstwą w projekcie jest *Model*. Jest to warstwa będąca pewną reprezentacją problemu [Zak06]. W zrealizowanym systemie składa się na nią szereg klas odpowiedzialnych za definicję typów danych wykorzystywanych przez aplikację (Rys. 4.2). Ich głównym zadaniem jest zamodelowanie rzeczywistości środowiska, w którym *TMT Analyzer* będzie znajdował swoje zastosowanie. Typy danych definiowane w warstwie *Modelu* są analogiczne do typów elementów zawartych w bazie danych.



Rysunek 4.2. Struktura pakietów warstwy Modelu.

W pakiecie **.org.hibernate.dialect** zawarto klasę odpowiedzialną za definicję konwersji typów zawartych w bazie danych na typy obiektowe wykorzystywane przez *Jawę*. Jej obecność jest konieczna, ponieważ *Hibernate* nie wspiera samoczynnie systemu zarządzania bazą danych *SQLite*. Definicja konwersji typów to fundamentalny proces w celu zrealizowania prawidłowego odwzorowania obiektowo-relacyjnego, przeprowadzanego przez platformę *Hibernate*. Ponadto, w pakiecie tym umieszczono obiektowe definicje funkcji wierszowych udostępnianych przez system *SQLite* w celu poprawnego ich mapowania na wyrażenia wykorzystywane podczas implementacji systemu.

Pakiet **.model** zawiera najważniejsze klasy w całej warstwie mające na celu zdefiniowanie typów obiektów wykorzystywanych przez system i zamodelowanie jego środowiska pracy. Wyróżniono tutaj interfejs **IEntity** implementowany przez każdą z 13 umieszczonych w pakiecie klas, mający na celu przypisanie unikalnego identyfikatora do wszystkich tworzonych obiektów. Każdy obiekt jest definiowany przez szereg cech zdefiniowanych przez klasę, której jest instancją. Ponadto, w plikach źródłowych klas umieszczono adnotacje wskazujące ściśle powiązanie danego obiektu do typów tabel zawartych w bazie danych. Dzięki nim, podczas korzystania z rekordów pochodzących z bazy danych w sposób automatyczny dokonywane jest mapowanie typów tych rekordów na typy obiektowe wykorzystywane przez *Jawę*. Przy definicji każdego pola klasy również występują

adnotacje, wskazujące na jego przypisanie do określonej cechy w tabeli zawartej w bazie danych oraz na krotność tego związku (Listing 4.2).

```

1 @Entity
2 @Table(name = "orders")
3 public class Order implements IEntity {
4     /* ... */
5     @JoinColumn(name = "from_point")
6     @ManyToOne
7     private Point fromPoint;
8 }

```

LISTING 4.2: Definicja typu obiektowego i jego powiązania z tabelą w bazie danych.

W pakiecie **.model.enums** zdefiniowano typy wyliczeniowe wykorzystywane podczas implementacji systemu. Ich zadaniem jest ułatwienie programowania i zapewnienie większej przejrzystości kodu ze względu na brak konieczności jego powielania i tworzenia nadmiarowych zmiennych. Typy wyliczeniowe zdefiniowane w pakiecie zawierają dostępne opcje dotyczące płci pacjenta, roli użytkownika w systemie lub części realizowanego testu łączenia punktów.

W pakiecie **.structs** umieszczono typy strukturalne wykorzystywane podczas implementacji systemu *TMT Analyzer*. Pozwalają one na definicję i ujednoczenie bardziej złożonych struktur, wykorzystywanych w trakcie tworzenia aplikacji i potrzebnych do jej prawidłowej pracy. Ich użycie ma dodatkowo na celu zapewnienie większej czytelności kodu poprzez redukcję liczby używanych obiektów podczas np. przeprowadzania testów statystycznych na sparowanych populacjach, a także umożliwienie poprawnego wyświetlania danych za pomocą obiektów udostępnianych przez technologię *JavaFX*.

W warstwie *Modelu* zawarto również plik konfiguracyjny mający postać dokumentu *XML* określający rodzaj połączenia z bazą danych i jednoznacznie wskazujący klasy zawarte w pakiecie **.model** podlegające mapowaniu obiektowo-relacyjnemu.

4.3.2 Widok

Warstwa *Widoku* jest odpowiedzialna za definicję sposobu, w jaki system wyświetla pewną część modelu w ramach interfejsu użytkownika [Zak06]. W warstwie tej umieszczono pliki *.fxml* określające własności i wymiary wszystkich okien wyświetlanych przez aplikację, a także zawartych w nich kontrolki użytkownika. Każdy taki plik źródłowy posiada definicję głównego panelu okna zawierającą jego unikalną (w ramach danego dokumentu) nazwę, minimalne i maksymalne wymiary oraz jednoznacznie określenie kontrolera odpowiedzialnego za przepływ sterowania w tym ekranie. Elementy umieszczone w danym oknie oraz ich właściwości są zdefiniowane w sekcji `<children> ... </children>` odpowiadającego dokumentu *.fxml*. Ponadto, każdy taki dokument posiada sekcję `<stylesheet>` w której programista powinien określić zastosowany styl *.css*.

W strukturze warstwy *Widoku* umieszczono również katalog, w którym zdefiniowano dwa style *.css* wykorzystywane w systemie. Jeden z nich określa styl wszystkich okien wyświetlanych przez aplikację oraz ich standardowych elementów, takich jak górna belka z przyciskami do minimalizacji i zamknięcia ekranu. Jego zastosowanie wpływa na jednolity wygląd wszystkich elementów graficznego interfejsu użytkownika, przez co jest on bardziej przyjazny w odbiorze. Drugi styl *.css* zawiera definicję wyglądu okienka wyboru daty, używanego np. podczas tworzenia nowej serii

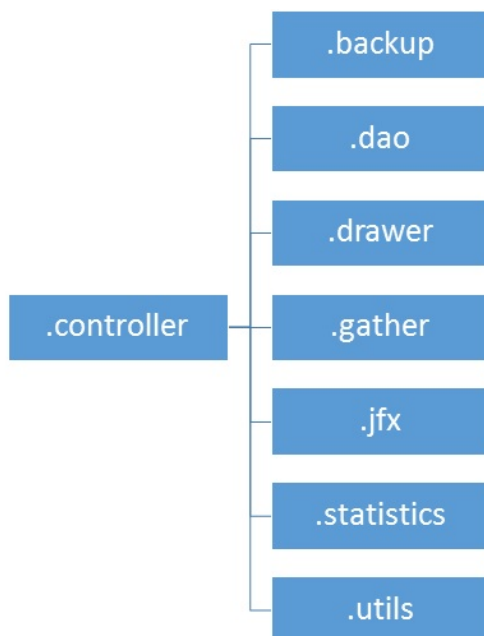
badań. Znajdują się w nim informacje dotyczące stylu okienka oraz charakterystyki zawartych w nim poszczególnych elementów. Jego fragment, określający właściwości strzałek nawigacji po kalendarzu przedstawiono na Listingu 4.3.

```
1  /* Ustalenie rozmiaru strzałek nawigacji w kalendarzu. */
2  .calendar-navigation-arrow {
3      -fx-padding: 4 3.5 4 3.5; /* ustalenie rozmiaru strzałki */
4      -fx-shape: "M 100 100 L 300 100 L 200 300 z"; /* kształt strzałki */
5      -fx-background-color: #ffffff; /* biały kolor jako jej tło */
6  }
```

LISTING 4.3: Fragment stylu DatePicker.css.

4.3.3 Kontroler

Kontroler jest ostatnią warstwą architektury zaimplementowanego systemu. Jego zadaniem jest obsługa przepływu sterowania między poszczególnymi oknami, a także realizacja całej funkcjonalności logiki biznesowej aplikacji. Zawarto w nim również pakiety klas *Javy* odpowiedzialne za przeprowadzenie podstawowych procesów mających na celu poprawną pracę systemu, takich jak pobieranie danych z tabletu lub kalibracja. Strukturę warstwy *Kontrolera* z podziałem na poszczególne pakiety przedstawiono na Rysunku 4.3.



Rysunek 4.3. Struktura pakietów warstwy Kontrolera.

W pakiecie **.backup** zdefiniowano klasy odpowiedzialne za przeprowadzenie procesu importu i eksportu danych. Funkcjonalności te zrealizowano przy użyciu dedykowanych metod udostępnianych przez specyfikację *JAXB*. Istnieje również możliwość realizacja kopii zapasowej bazy danych polegającej na przeprowadzeniu operacji eksportu całej jej zawartości do zewnętrznego pliku.

Pakiet **.dao** posiada definicje klas zaimplementowanych w taki sposób, aby tworzyły one komponenty *Data Access Object*, dostarczające jednolity interfejs do komunikacji między aplikacją a źródłem danych. Wyróżniono tutaj generyczną klasę abstrakcyjną **GenericDao** zawierającą szablony metod odpowiedzialnych za pobranie jednego lub kilku elementów z bazy, ich filtrację i modyfikację. Klasa ta jest rozszerzana przez pozostałe klasy umieszczone w pakiecie, które dodatkowo udostępniają wiele innych metod operujących na danych, w zależności od pełnionej funkcjonalności.

W pakiecie **.drawer** umieszczono klasy odpowiedzialne za przeprowadzenie odwzorowania toru kreślenia pacjenta zarówno w trybie on-line, jak i podczas przeglądania wyników jego badań. Wyróżniono tutaj interfejs **Drawer** w którym umieszczono nagłówki metod odpowiedzialnych za przygotowanie kanwy (ang. *canvas*) umożliwiającej rysowanie oraz realizację procesu wizualizacji. Pozostałe klasy znajdujące się w pakiecie implementują ten interfejs w sposób zależny od ich zaplanowanej funkcjonalności.

Pakiet **.gather** jest odpowiedzialny za realizację pobierania danych z tabletu. Klasy w nim zdefiniowane posiadają metody, które odczytują zawartość struktur *htd* pochodzących z tabletu i na ich podstawie generują pakiety, które następnie są wprowadzane do dedykowanej tabeli w bazie danych. Przeprowadzenie tego procesu odbywa się przy pomocy biblioteki *JPen*, udostępniającej metody weryfikujące położenie pióra na tablecie, a także jego nacisk i kąt odchylenia.

W pakiecie **.jfx** umieszczono kontrolery okien wyświetlanych przez system, określających jego interakcje z użytkownikiem. Każdy taki kontroler jest przypisany do konkretnego okna systemu i posiada również metody odpowiedzialne za jego inicjalizację i zamknięcie. Kluczową klasą w pakiecie jest **Tablet**, która definiuje przepływ sterowania między poszczególnymi oknami, a także jest punktem startowym aplikacji. Umieszczono w niej również metody definiujące sposób wyświetlania ekranów, zależny od funkcjonalności przez nie udostępnianych.

Następnym pakietem zdefiniowanym w warstwie *Kontrolera* jest **.statistics**. Zadaniem klas w nim umieszczonych jest pobranie wstępnie przetworzonych danych dotyczących konkretnych badań i przeprowadzenie weryfikacji postępów w wynikach badań danego pacjenta, spowodowanych przeprowadzeniem zabiegu lub podaniem leku. Ponadto, pakiet **.statistics** udostępnia również metody odpowiedzialne za sprawdzenie rozkładu cech danej populacji i w zależności od niego realizujące odpowiednie testy statystyczne na dwóch sparowanych populacjach. Celem tych testów jest zbadanie istotnych statystycznie różnic między tymi populacjami.

Pakiet **.utils** zawiera klasy odpowiedzialne za pozostałe funkcjonalności systemu. Zaliczono do nich przeprowadzenie kalibracji tabletu, szyfrowanie parametrów logowania użytkowników oraz generowanie raportu zawierającego wyniki analizy danych w postaci pliku PDF. Zdefiniowano tutaj również kilka klas ułatwiających proces implementacji systemu, wpływających na czytelność kodu, które udostępniają metody operujące na liczbach lub łańcuchach znaków.

Rozdział 5

Opis implementacji i realizacji projektu

W niniejszym rozdziale omówiono zagadnienia związane z procesem implementacji projektu *TMT Analyzer*. Najpierw, przedstawiono informacje ogólne dotyczące zastosowanej technologii programistycznej oraz przyjętych standardów kodowania. Następnie, omówiono wybrane, najciekawsze zagadnienia implementacyjne wpływające na logikę biznesową projektu. W dalszej części rozdziału scharakteryzowano przebieg realizacji całego przedsięwzięcia i zastosowany sposób organizacji pracy, a także opisano sposoby komunikacji w zespole. W podsumowaniu rozdziału odniesiono się do napotkanych trudności i doświadczeń zdobytych podczas pracy nad implementacją systemu.

5.1 Informacje ogólne i standard kodowania

Implementacja systemu została przeprowadzona za pomocą zintegrowanego środowiska programistycznego *Eclipse*, przy użyciu technologii i obiektowego języka programowania *Java*. Podczas realizacji projektu przyjęto za dobrą praktykę używanie standardu kodowania i formatowania plików źródłowych opracowanego przez firmę *Oracle*. Zawarto w nim informacje dotyczące sposobu nazewnictwa klas, pól oraz metod w nich zawartych, a także używania komentarzy i specyficznych bloków kodu. Dokument ten również opisuje szereg dobrych praktyk programistycznych, dotyczących organizacji zawartości i odpowiedniego nazewnictwa plików źródłowych.

Do nazwania klas oraz interfejsów zalecane jest używanie wielkiej litery na początku nazwy oraz do każdego wewnętrznego słowa w nazwie. Nazwy pól i metod powinny być zapisywane małą literą, a każde ich wewnętrzne słowa powinny rozpoczynać się wielką literą. Nazwy stałych używanych w programie powinny być zapisywane w całości wielkimi literami [Ora99].

Standard kodowania zaleca stosowanie spacji po słowach kluczowych, po których dodatkowo występują nawiasy. Ponadto, wskazane jest wstawianie spacji przed nawiasami otwierającymi zawartość bloku kodu. Nawiasy te powinny być zapisywane na końcu aktualnej linii. Ich stosowanie jest obowiązkowe, nawet jeżeli w ciele danego bloku kodu występuje tylko jedna instrukcja. W celu zamknięcia bloku, stosuje się nawias zamykający, umieszczany w nowej linii.

Standard kodowania definiuje również maksymalną zalecaną długość linii kodu, nieprzekraczającą 80 znaków. Niestosowanie tej zasady może powodować nieprawidłową obsługę pliku źródłowego przez niektóre narzędzia i terminale [Ora99]. Używanie komentarzy jest wskazane w celu opisania intencji programisty, podsumowania algorytmu lub wskazania logicznego przepływu. Zaleca się ich wyraźną separację od wyrażeń przy użyciu odpowiednio dużego przesunięcia.

```
1 public class ImageSprite implements ISprite {
2     public boolean doneOrNot;
3     public void DrawImage() {
4         while (true) {
5             if (done) {
6                 statement1;           /* ... */
7             } else {
8                 statement2;         /* ... */
9             }
10        }
11    }
12 }
```

LISTING 5.1: Przykład nazywania klas pól i metod oraz stosowania bloków kodu

5.2 Wybrane zagadnienia implementacyjne

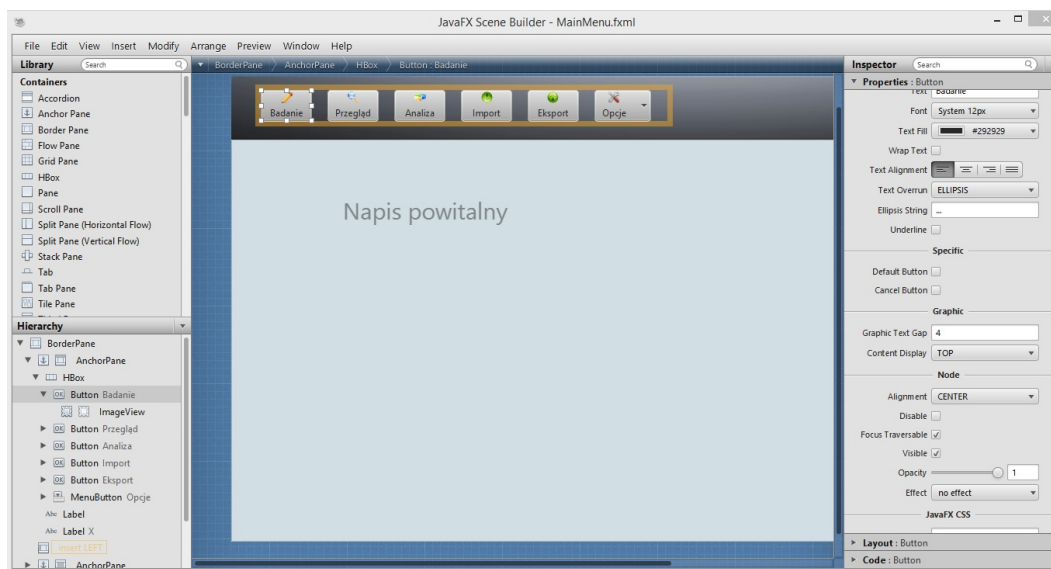
5.2.1 Pobieranie danych z tabletu

Chcąc zaimplementować system wspierający przeprowadzanie testu łączenia punktów oparty na współpracy komputera z tabletem graficznym, należy przede wszystkim umożliwić odpowiednią komunikację tego systemu z urządzeniem. Do implementacji pobierania danych z tabletu wykorzystano bibliotekę *JPen*, udostępniającą metody odpowiedzialne za odczytywanie współrzędnych pióra na tablecie oraz pozostałych związanych z nim parametrów zawartych w strukturze *htd*. Metody te zaimplementowano w oparciu o wzorzec projektowy *Listener*, polegający na wyróżnieniu dwóch podstawowych typów obiektów: obserwowanego i obserwatora. Zgodnie z budową wzorca, kiedy stan obiektu obserwowanego się zmienia, wywołuje on metodę, która wysyła powiadomienia do wszystkich zarejestrowanych obserwatorów [Hol05]. Za pomocą tych metod, w przypadku kontaktu pióra z powierzchnią tabletu, następuje pobieranie pakietów danych z częstotliwością 200 Hz (1 pakiet na 5 milisekund). Dane te są wprowadzane do dedykowanej tabeli w bazie danych. Ich przechowywanie ma na celu umożliwienie przeprowadzenia wstępnej analizy i agregacji tych danych, a ponadto informacje dotyczące współrzędnych odczytanych z tabletu służą do wizualizacji toru kreślenia pacjenta. Zaimplementowano również możliwość pobrania tylko jednego punktu, wykorzystywaną podczas przeprowadzania kalibracji tabletu. W tym przypadku, po powiadomieniu obiektu będącego obserwatorem, metoda podlega zakończeniu. Moduł odpowiedzialny za realizację pobierania danych umieszczono w warstwie Kontrolera.

5.2.2 Graficzny interfejs użytkownika

Stworzenie interfejsu graficznego systemu *TMT Analyzer* wymagało wyboru odpowiednich narzędzi i komponentów, a także określenia przepływu sterowania w programie i interakcji z użytkownikiem. W tym celu, gdy zaprojektowano poszczególne moduły i funkcjonalności aplikacji, zdecydowano się na utworzenie prototypu cyfrowego za pomocą środowiska *Lumzy*, ułatwiającego wizualizację pomysłów i koncepcji dotyczących interfejsu użytkownika. Głównym zadaniem za-

projektowanego interfejsu jest umożliwienie korzystania z zaimplementowanych funkcji systemu w intuicyjny sposób. Poszczególne ekrany wyświetlane przez system zaprojektowano za pomocą narzędzia *JavaFX Scene Builder* udostępnianego przez platformę *JavaFX*. Pozwala ono na utworzenie plików w formacie *.fxml* zawierających definicje rozmieszczenia elementów okien, ich właściwości, kolorystyki i rozmiarów. Kod źródłowy tych plików jest tworzony w sposób dynamiczny, a użytkownik programu posiada dostęp do wygodnego trybu projektowania (Rys. 5.1).



Rysunek 5.1. Okno projektowania w narzędziu Scene Builder

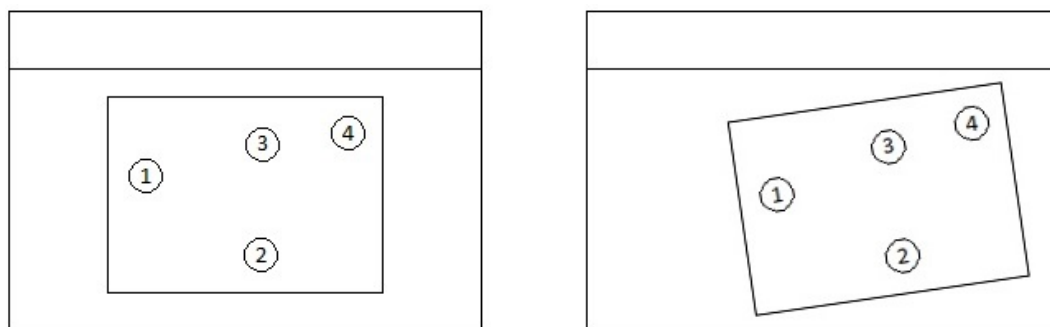
Do utworzenia poszczególnych okien zastosowano technologię *JavaFX*, natomiast definicję odpowiedniego przepływu sterowania między tymi oknami zawarto w warstwie Kontrolera. Każdy ekran posiada odpowiadającą klasę, której zadaniem jest udostępnianie metod obsługujących wszystkie zdefiniowane zdarzenia występujące w danym oknie. Zdefiniowano tam również sposoby interakcji między poszczególnymi elementami każdego okna, obsługę błędów użytkownika, a także wywołania metod inicjalizujących dane okno.

Podstawową i najważniejszą klasą z punktu widzenia interfejsu graficznego jest klasa *Tablet*, zawierająca metody odpowiedzialne za uruchomienie i inicjalizację systemu, definicję sposobów przechodzenia między poszczególnymi oknami oraz ustanowienie połączenia z bazą danych, niezbędnego do umożliwienia użytkownikowi dostępu do interesujących go danych.

5.2.3 Kalibracja położenia kartki na tablecie

Różnica w rozmiarach arkusza z testem łączenia punktów i powierzchni tabletu spowodowała potencjalne problemy wynikające z ewentualnego przemieszczenia i przekręcenia kartki na urządzeniu (Rys. 5.2). Kwestia ta wymusiła konieczność opracowania metody weryfikującej względne położenie testu na tablecie. Metoda ta jest niezbędna do poprawnego działania wielu ważnych funkcjonalności systemu, takich jak wizualizacja toru kreślenia lub analiza danych.

W celu rozwiązania potencjalnych problemów związanych z niepoprawnym przetwarzaniem danych uzyskanych z tabletu, zaimplementowano metodę przeprowadzającą jego kalibrację. Jej zadaniem jest określenie względnej pozycji kartki i dostosowanie do niej funkcji przetwarzających.



Rysunek 5.2. Przykład wzorcowego i niepoprawnego montażu kartki na tablecie.

W tym celu, na arkuszu z testem umieszczono trzy punkty kalibracyjne, które przed każdym badaniem użytkownik powinien, zgodnie z kolejnością ich numeracji, wskazywać piórem. W taki sposób system otrzymuje informacje o ich położeniu względem tabletu i może łatwo dostosować funkcje odpowiedzialne za przetwarzanie danych pochodzących z testu.

Pewnym problemem związanym z implementacją procesu kalibracji okazał się brak odpowiednich sterowników i bibliotek systemowych odpowiedzialnych za współpracę z tabletem na komputerach dwóch członków zespołu. Sytuacja ta skutecznie utrudniała testowanie dodawanych modułów aplikacji, jednak po pewnym czasie znaleziono odpowiednie rozwiązanie, polegające na wprowadzeniu odpowiednich plików do katalogu systemowego i wszyscy członkowie grupy mieli możliwość swobodnego korzystania z tabletu na swoich komputerach.

5.2.4 Odwzorowywanie toru kreślenia

Jedną z ważniejszych funkcjonalności systemu jest wizualizacja toru kreślenia pacjenta, realizowana w trybie *on-line* podczas przeprowadzania badania. Jej opracowanie i implementacja miały na celu umożliwienie osobie nadzorującej realizację testu jego kontrolę i wychwytywanie na bieżąco błędów pacjenta polegających na przejściu do nieprawidłowego punktu. Dodatkowo, zdecydowano się wykorzystać tę funkcjonalność w module przeglądu badań. Wówczas, użytkownik przeglądając wyniki danego pacjenta, również posiada dostęp do odwzorowania jego toru kreślenia. W celu zaimplementowania tej funkcji, w specjalnie do tego przeznaczonym pakiecie klas *Java* utworzono interfejs **Drawer** zawierający sygnatury metod odpowiedzialnych za przygotowanie okna systemu polegające na utworzeniu kanwy umożliwiającej rysowanie oraz umieszczeniu na niej punktów z określonej części testu.

Udostępnienie tej funkcjonalności polegało na stworzeniu klasy implementującej interfejs **Drawer**. W celu weryfikowania błędów pacjenta, zaimplementowano klasę **Guardian**, która na bieżąco sprawdza kolejność osiągniętych punktów i w przypadku zlokalizowania nieprawidłowego przejścia wywołuje metodę, której zadaniem jest oznaczenie na ekranie danego punktu jako błędnie połączonego. Klasa ta jest również odpowiedzialna za inicjalizację procesu rysowania i jego zakończenie.

Podczas implementacji odwzorowywania toru kreślenia wyróżniono również klasę **Projector**, której zadaniem jest dostosowanie całego modułu do ustawień i kalibracji tabletu. Za jej pomocą rozwiązano problemy wynikające z nieprawidłowego montażu kartki z testem na urządzeniu. Klasa ta udostępnia metody odpowiedzialne za przeliczanie współrzędnych punktów względem wykonanej przez użytkownika kalibracji oraz dostosowanie funkcji przetwarzających dane do położenia testu

na tablecie. Opracowanie metody odpowiedzialnej za poprawne zdefiniowanie ewentualnego kąta obrotu kartki na tablecie w oparciu o posiadane informacje dotyczące współrzędnych punktów kalibracyjnych wymusiło zamodelowanie problemu matematycznego i podanie jego rozwiązania. W ten sposób przypadkowe przesunięcie arkusza lub jego niewielkie obrócenie nie wpływają na jakość wyznaczanych parametrów ani na wizualizację toru kreślenia pacjenta.

5.2.5 Moduł statystyczny

Jednym z dwóch kluczowych zadań, jakie postawiono przed systemem *TMT Analyzer*, poza automatyzacją procesu przeprowadzania badań, jest umożliwienie szczegółowej analizy danych pochodzących z testów. Zadanie to zrealizowano dwuetapowo. Najpierw, surowe dane pochodzące z tabletu poddawane są wstępnemu przetworzeniu, celem wyznaczenia na ich podstawie podstawowych parametrów dotyczących przebiegu badania. Następnie, użytkownik systemu, uruchamiając moduł właściwej analizy danych, może przeprowadzić na nich testy statystyczne mające na celu stwierdzenie istotności różnic między dwoma, sparowanymi ze sobą populacjami (np. pacjentów z Krakowa i pacjentów z Poznania). Moduł ten udostępnia również możliwość weryfikacji różnic w wynikach badań dwóch pacjentów, a także sprawdzenia postępów pojedynczego pacjenta, wynikających z zastosowania kuracji lekowej lub przeprowadzenia zabiegu.

Do przeprowadzenia wstępnego przetwarzania danych pochodzących z tabletu, zaimplementowano klasę `Processor`, znajdującą się w pakiecie `.utils` w warstwie *Kontrolera*. Pakiety pobierane z tabletu są tworzone w odpowiednie struktury i umieszczane w dedykowanej tabeli w bazie danych. Po zakończeniu każdego badania, system wyświetla okno, za pomocą którego użytkownik może zapisać wyniki badania lub je odrzucić. W przypadku wyboru zapisania przebiegu testu, system tworzy nowy obiekt klasy `Processor` i za jego pomocą wywołuje metodę odpowiedzialną za przeprowadzenie wstępnej analizy danych. Polega ona na wyznaczeniu poniższych parametrów charakteryzujących przebieg każdego badania:

- całkowity czas trwania testu,
- całkowita długość linii kreślenia pacjenta,
- średnia i maksymalna prędkość kreślenia oraz odchylenie standardowe prędkości,
- średnie i maksymalne przyspieszenie kreślenia oraz odchylenie standardowe przyspieszenia,
- średni nacisk pióra na tablet i odchylenie standardowe nacisku,
- liczba oderwań pióra od tabletu,
- liczba popełnionych przez pacjenta błędów,
- łączny czas przebywania w punktach,
- czas łączenia punktów,
- całkowity dystans od wygładzonej linii kreślenia,
- widma drżeń prędkości kreślenia, przyspieszenia i siły nacisku.

Całkowity czas trwania testu jest wyznaczany na podstawie momentu pobrania pierwszego i ostatniego pakietu pochodzącego z tabletu. Długość linii kreślenia pacjenta wyznaczono przy użyciu analitycznego wzoru na odległość między dwoma punktami, których dokładne współrzędne są znane. Na podstawie tych dwóch parametrów wyznaczono prędkości i przyspieszenia ruchu ręki pacjenta. Nacisk pióra na tablet, a na jego podstawie liczbę oderwań, iteracyjnie odczytywano z pakietów pochodzących z tabletu. Łączny czas przebywania w punktach, czyli tzw. czas na-

mysłu, oraz liczbę popełnionych przez pacjenta błędów wyznaczono na podstawie współrzędnych punktów umieszczonych na arkuszu z testem. W celu wyznaczenia całkowitej odległości od wygładzonej linii kreślenia, zastosowano następujące rozwiązanie, opierające się na metodzie regresji liniowej:

1. poprowadzenie prostej między i -tym i $(i+5)$ -tym punktem na linii kreślenia.
2. posumowanie odległości od punktów pośrednich do wyznaczonej prostej.

Widma drzeń określonych charakterystyk wyznaczono przy użyciu zewnętrznej biblioteki udostępniającej odpowiednie metody. Ich parametrami były statyczne tablice zawierające dane podlegające przetworzeniu. Metody te zaimplementowano na podstawie *Szybkiej Transformacji Fouriera* (ang. *Fast Fourier Transformation (FFT)*).

Dane przetworzone są zapisywane do dedykowanej tabeli w bazie danych. Na ich podstawie realizowane są operacje mające na celu przeprowadzenie właściwej analizy wyników badań.

Moduł analizy danych udostępnia możliwość parowania pacjentów, której celem jest odpowiednie przypisanie wyników badań, na podstawie którego użytkownik chciałby przeprowadzić testy statystyczne. Tak sparowane populacje są podstawą do weryfikacji istotnych różnic między wynikami ich badań. Zaimplementowano również możliwość testowania wyników jednego pacjenta, mającą na celu sprawdzenie postępów wynikających z zastosowanej kuracji lekowej lub zabiegu. Analogicznie, istnieje możliwość zweryfikowania różnic między wynikami wybranych przez użytkownika dwóch pacjentów.

W przypadku badania postępów w wynikach określonego pacjenta, następuje porównanie wartości parametrów statystycznych wyznaczonych na podstawie badań przeprowadzonych przed i po zastosowaniu terapii lub zabiegu. Następnie, system wylicza dokładne różnice i określa, w jaki sposób i o ile wartości poszczególnych parametrów zmieniły się wskutek przeprowadzonego leczenia. W taki sam sposób zrealizowano porównywanie wyników badań dwóch pacjentów. Wówczas, również następuje wskazanie dokładnej różnicy parametrów wyznaczonych na podstawie zarejestrowanych przebiegów testów.

W przypadku testowania różnic między dwiema populacjami, zdecydowano się na podejście dwuetapowe. Po uprzednim sparowaniu populacji, następuje zbadanie rozkładów ich poszczególnych cech. Proces ten zrealizowano za pomocą **testu Lillieforsa**, dedykowanego do określania, czy rozkład danej cechy jest normalny. W zależności od wyniku przez niego wygenerowanego, przeprowadza się **test U Manna-Whitneya** lub **test t-Studenta**. Pierwszy z nich jest nieparametrycznym testem do sprawdzenia, czy następuje istotna różnica między poszczególnymi wartościami parametrów próbek pobranych z dwóch niezależnych populacji. Odrzucenie hipotezy zerowej oznacza stwierdzenie, że pod względem danej cechy dwie populacje różnią się od siebie istotnie. Test **t-Studenta** jest parametrycznym testem stosowanym również w celu zbadania różnic między dwoma próbkami. Implementacja testów **t-Studenta** i **U Manna-Whitneya** opierała się także na bibliotece `commons.math`, natomiast do zaimplementowania testu **Lillieforsa** posłużono się biblioteką `jdist.lib`. Parametrami wejściowymi do metod testujących były wartości charakterystyk testowanych populacji. Warto zauważyć, że badanie różnic między populacjami polega na przetestowaniu każdej cechy osobno. Parametrem wyjściowym metod testujących jest odpowiedź, czy przy danym poziomie istotności nastąpiła istotna statystycznie różnica między dwoma populacjami. Poziom istotności ustalono na wartość 0.05, a także zaimplementowano możliwość obliczenia p-wartości dla danego testu.

5.2.6 Import i eksport danych

Podczas projektowania systemu *TMT Analityzer* zdecydowano się na utworzenie modułu importu i eksportu danych w celu umożliwienia użytkownikom pracy na zewnętrznych plikach zawierających istotne dla nich dane medyczne, a także z powodu przyzwyczajęń grupy docelowej systemu oraz możliwości łatwego przenoszenia grup badań. Podczas implementacji tych funkcjonalności, skorzystano ze specyfikacji *JAXB*, służącej do mapowania obiektów *Javy* na dane przechowywane w postaci dokumentu *XML*.

Import danych polega na utworzeniu obiektu klasy *Unmarshaller*, której zadaniem jest deserializacja danych zawartych w dokumencie *XML* do obiektu wykorzystywanego przez *Jawę*. Proces ten jest realizowany za pomocą metody *unmarshal*, której parametrem jest ścieżka do pliku zewnętrznego zawierającego dane do importu (Listing 5.2). Implementacja procesu eksportu polega na utworzeniu obiektu klasy *Marshaller*, dzięki któremu wykonano metodę *marshal*. Parametrami tej metody są ścieżka do tworzonego pliku zewnętrznego oraz zbiór danych, które użytkownik chciałby wyeksportować. Klasy *Marshaller* i *Unmarshaller* są udostępniane przez specyfikację *JAXB*, stąd nie było konieczności ich implementacji.

```
1 public Integer importData(File file) throws JAXBException {
2     /* ... */
3     Unmarshaller unmarshaller = JAXBContext.createUnmarshaller();
4     if (file.exists()) {
5         DataContainer data = (DataContainer) unmarshaller.unmarshal(file);
6         /* zapis do bazy */
7         return data.getData().size();
8     }
9     return null;
10 }
```

LISTING 5.2: Metoda importująca dane z pliku zewnętrznego

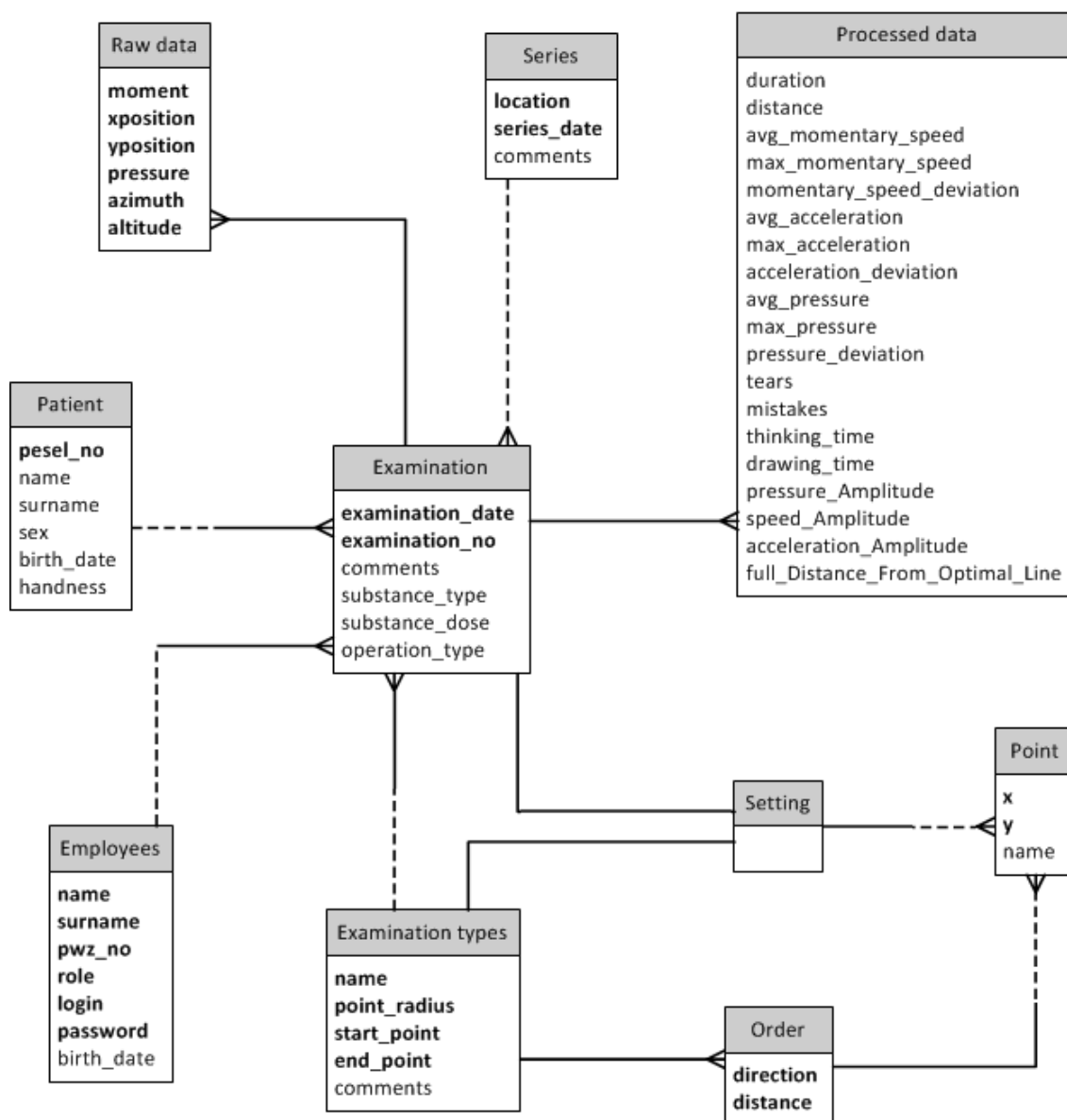
5.2.7 Metody przechowywania danych

Dane medyczne są szczególnie wrażliwe, dlatego na opracowaniu efektywnych metod ich przechowywania położono duży nacisk. Podczas projektowania systemu rozważano kilka potencjalnych rozwiązań.

Pierwszą opcją, która samoistnie przychodzi na myśl, jest zastosowanie dedykowanej bazy danych. Główną korzyścią wynikającą z takiego rozwiązania jest łatwość obsługi z poziomu kodu, wbudowana podstawowa kontrola poprawności danych, a także efektywność w przetwarzaniu złożonych zależności. Z drugiej strony, bazy danych najczęściej charakteryzuje wysokie zapotrzebowanie na zasoby oraz kłopotliwa instalacja. Dodatkowo, nie zapewniają one trywialnego przenoszenia danych, co stanowiłoby pewien problem podczas eksportowania wszystkich wyników zawartych w bazie danych. Drugą opcją, którą rozważano, było zastosowanie konwencjonalnych plików mających budowę dokumentu *XML*. Rozwiązanie takie zapewniałoby łatwość przenoszenia niewielkich grup badań. Natomiast, utrzymywanie danych w postaci systemu plików wpływa na ograniczoną funkcjonalność i powoduje konieczność utrzymania odpowiedniego porządku danych w strukturze katalogów.

Projektując program *TMT Analyzer* zdecydowano się na zastosowanie systemu zarządzania bazą danych *SQLite*. Cechuje go przede wszystkim niewielkie zapotrzebowanie na zasoby i brak konieczności instalacji na komputerze użytkownika, co rozwiązuje dwa największe problemy napotykane podczas stosowania baz danych. Ponadto, chcąc wykorzystać atuty utrzymywania danych w postaci plików zewnętrznych, zdecydowano się na zaimplementowanie modułu importu i eksportu danych. W taki sposób osiągnięto kompromis między dwoma rozważanymi rozwiązaniami, jednocześnie zachowując ich największe zalety.

W celu poprawnej organizacji przechowywanych danych utworzono szereg tabel odpowiedzialnych za ich odpowiednią strukturalizację (Rys. 5.3).



Rysunek 5.3. Schemat zastosowanej bazy danych.

Centralnym elementem bazy danych jest tabela **Examination**. Zawiera ona wszystkie informacje dotyczące przeprowadzonego badania, tj. data i numer badania oraz rodzaj przeprowadzonej przed nim kuracji medycznej. Dodatkowo, tabela zawiera odniesienia do innych struktur celem

zgromadzenia jak największej ilości danych dotyczących danego badania. W ten sposób przypisano do niego również konkretnego pacjenta, osobę nadzorującą badania, serię badań i konfigurację dotyczącą kalibracji. Ponadto, każde badanie posiada odniesienie do rekordu zawierającego przetworzone przez system dane. Zadaniem tabeli *Point* jest przechowywanie współrzędnych punktów umieszczonych na testach oraz zdefiniowanych podczas każdego procesu kalibracji. Posiada ona ścisły związek z tabelą *Order*, przechowującą dane dotyczące poszczególnych przejść między punktami, tj. punkt początkowy i końcowy przejścia, jego długość i kierunek.

5.3 Przebieg prac nad projektem

W niniejszym podrozdziale przedstawiono szczegóły dotyczące przebiegu procesu projektowania i implementacji systemu *TMT Analyzer*. Poruszono wszystkie aspekty prac nad projektem oraz zastosowany sposób rozwiązania typowych problemów i kwestii organizacyjnych. Ponadto, wyszczególniono również informacje dotyczące harmonogramu prac nad aplikacją.

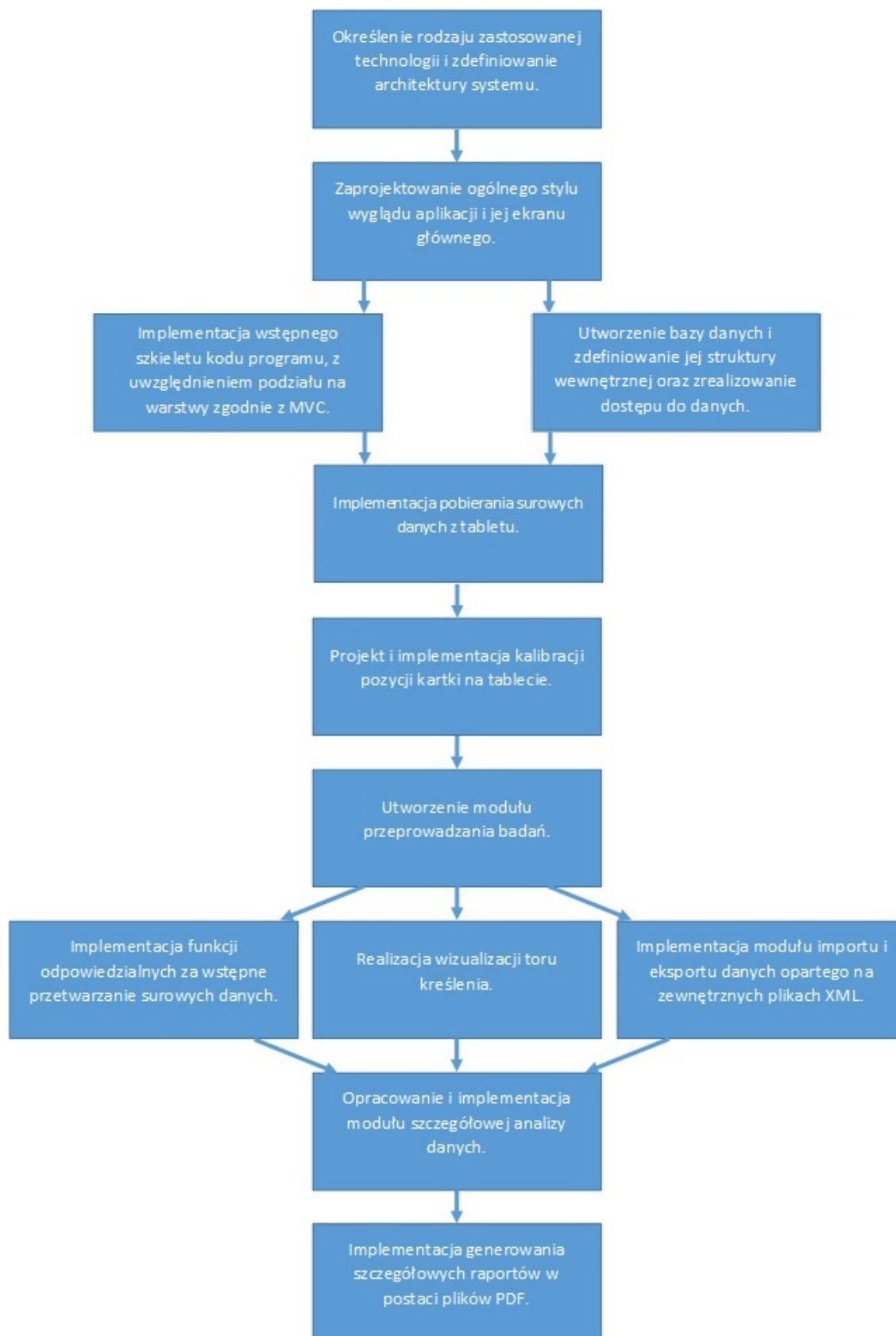
5.3.1 Informacje ogólne

Implementację systemu *TMT Analyzer* podzielono na szereg zadań, realizowanych pojedynczo przez członków zespołu lub w zespołach dwuosobowych w zależności od stopnia ich zaawansowania oraz wymaganego nakładu pracy. Listę zadań aktualizowano na bieżąco z powodu ciągłego występowania nowych problemów lub drobnych zmian koncepcji. Do prawidłowego utrzymywania listy zadań wykorzystano narzędzie internetowe *Teambox*, ułatwiające pracę w zespole. Warto podkreślić, że realizacji każdej nowej funkcjonalności systemu towarzyszyło utworzenie okna interfejsu graficznego, którego zadaniem jej jest udostępnienie użytkownikowi. Ponadto, po ukończeniu wszystkich prac nad projektem przeprowadzono liczne refaktoryzacje mające na celu poprawienie jakości kodu oraz jego czytelności. Szczegółową kolejność zrealizowanych zadań przedstawiono na Rysunku 5.4.

5.3.2 Harmonogram prac i komunikacja w zespole

Prace nad systemem *TMT Analyzer* zostały rozpoczęte we wrześniu 2013 roku wraz z podjęciem pierwszych konkretnych decyzji dotyczących procesu projektowania i implementacji projektu. Określono wtedy także wstępne zadania przeznaczone do realizacji w przeciągu następujących dwóch tygodni. Ponadto, poczyniono kroki mające na celu organizację prac nad projektem oraz ułatwienie komunikacji w zespole.

Proces implementacji podzielono na szereg zadań, które dodatkowo zgrupowano w przyrosty. Ponadto, od samego początku położono duży nacisk na odpowiednią komunikację w zespole, co miało na celu prawidłowy przepływ informacji między jego członkami. Podczas cotygodniowych spotkań, każdy członek zespołu (lub para realizująca dane zadanie) informował pozostałych o stopniu zaawansowania prac i ewentualnych problemach z nimi związanych. Wówczas, przedyskutowywano je wspólnie w grupie tak, aby uzyskać jednoznaczne stanowisko w danej kwestii lub sposób rozwiązania problemu. Dodatkowo, organizowano spotkania z opiekunem zespołu odbywające się regularnie co dwa tygodnie, mające na celu rozwiązanie większych trudności związanych z koncepcjami dotyczącymi systemu lub zasięgnięcie opinii na temat możliwych dodatkowych wymagań.



Rysunek 5.4. Schemat obrazujący kolejność realizowanych zadań.

W drugiej części okresu implementacji systemu grupa regularnie spotykała się w celu wspólnego programowania i testowania aplikacji. Ponadto, w celu wspierania efektywnej pracy w zespole, skorzystano z systemu kontroli wersji, który umożliwiał automatyczne powiadamianie o każdej zmianie w projekcie. W taki sposób członkowie zespołu byli skutecznie i szybko informowani o wszystkich modyfikacjach kodu programu. System ten umożliwiał również utrzymywanie historii zmian w projekcie i ich wygodną analizę.

Termin ukończenia procesu tworzenia systemu *TMT Analyzer* wstępnie ustalono na drugą połowę stycznia 2014 roku. Do tego czasu zaimplementowano wszystkie funkcjonalności określone za pomocą wymagań funkcjonalnych, a także przetestowano aplikację, w celu weryfikacji poprawności jej działania oraz przetwarzania przez nią uzyskanych danych medycznych.

5.4 Zebrane doświadczenia

Realizacja projektu pozwoliła na zapoznanie się z dotychczas nieznanymi członkom zespołu, a jednocześnie bardzo ułatwiający pracę technologiami. Konieczność ich poznania bardzo wpłynęła na zakres wiedzy autorów i ich swobodę w pracy nad projektami informatycznymi. Takie doświadczenie z pewnością pozwoli na szersze spojrzenie na realizację podobnych problemów i zadań w przyszłości. Kluczowa podczas projektowania graficznego interfejsu użytkownika systemu *TMT Analyzer* znajomość technologii *JavaFX* może okazać się przydatna podczas projektów opartych na idei bogatych aplikacji internetowych. Jest to technologia umożliwiająca tworzenie lekkich programów użytkowych, których przeznaczeniem są np. telefony komórkowe, obsługujące środowisko *Javy*. Praca z narzędziem *Apache Maven* automatyzującym proces budowy projektu jest zjawiskiem powszechnym w profesjonalnych zespołach programistycznych. W związku z tym, doświadczenie zdobyte podczas tworzenia systemu *TMT Analyzer* pozwoli w przyszłości na efektywniejszą i swobodniejszą pracę z tym narzędziem. Dodatkowo, zastosowanie technologii *Hibernate* i *JAXB* w zrealizowanym projekcie wpłynie na swobodę członków zespołu podczas programowania modułów opartych na obsłudze danych z poziomu aplikacji. Ciekawym doświadczeniem okazała się być również praca z tabletem graficznym. Poznanie zakresu jego możliwości, sposobów wykorzystania, a także tworzenia dedykowanego na niego oprogramowania sprawiło, że wiedza członków zespołu dotycząca tego typu urządzeń została znacznie poszerzona.

Realizacja systemu *TMT Analyzer* pozwoliła na sprawdzenie się w pracy zespołowej. Doświadczenie zdobyte podczas projektu grupowego będzie w przyszłości stanowiło ważny element dla członków zespołu. Właściwy podział zadań i obowiązków pomiędzy programistami umożliwił sprawne ich wykonanie. Wspólną pracę zdecydowanie usprawnił system kontroli wersji, umożliwiający śledzenie na bieżąco zmian w kodzie źródłowym projektu. Odpowiednia wiedza na temat takich systemów, nabyta podczas realizowania projektu, może okazać się kluczowa podczas realizacji przyszłych aplikacji. Ponadto, wspólne pokonywanie nowych przeszkód i radzenie sobie z problemami sprawiło, że cały proces projektowania i implementacji systemu był bardzo pozytywnym doświadczeniem.

Ostatnim elementem wyniesionym z pracy nad realizacją systemu było nabycie świadomości, że informatyka faktycznie posiada duży wpływ na ludzką pracę i sprawia, że staje się ona bardzo efektywna. Poczucie, że praca zespołu zostanie przez kogoś wykorzystana i doceniona, sprawiło, że implementacja projektu dostarczyła dużej radości i satysfakcji jego autorom.

Rozdział 6

Użytkowa strona aplikacji

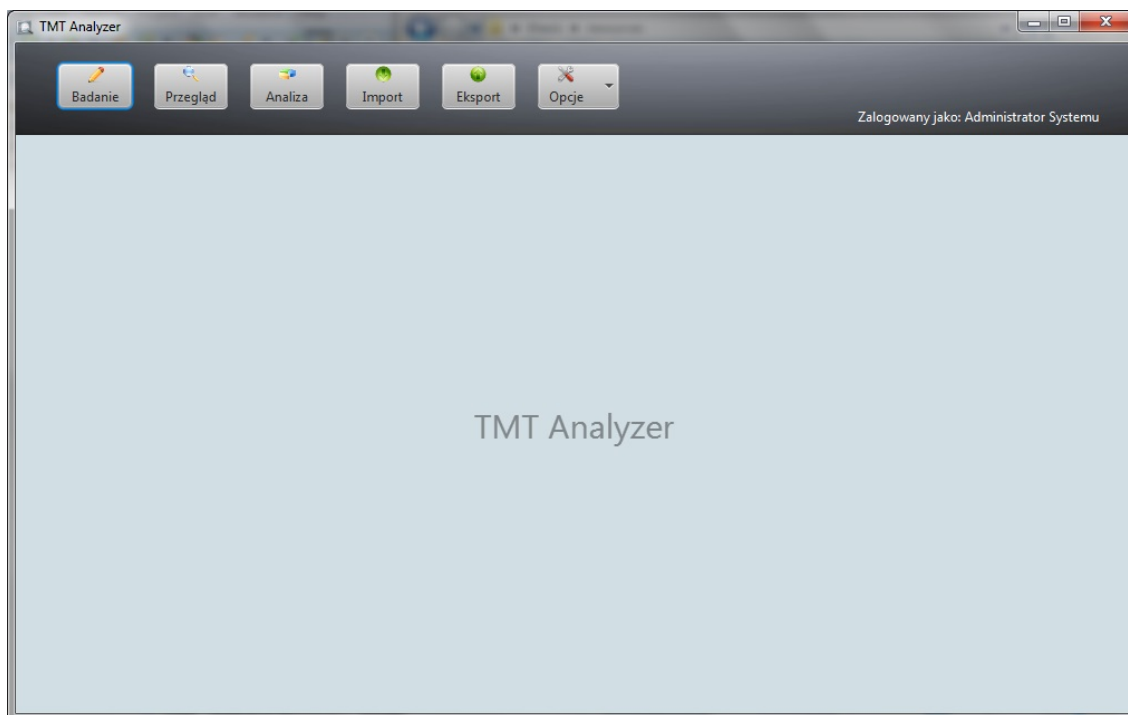
Graficzne środowisko systemu *TMT Analyzer* stworzono w oparciu o okna stosowane w systemach rodziny *Windows* (ang. *Windows Forms*) oraz o technologię *JavaFX*, dzięki której nadano mu elegancki i ergonomiczny wygląd. Interfejs użytkownika opracowywano stopniowo, co oznacza, że wraz z tworzeniem nowej funkcjonalności, projektowano okno systemu, którego zadaniem było jej udostępnienie. Skorzystano również z narzędzia *Lumzy*, umożliwiającego utworzenie prototypu cyfrowego, dzięki czemu proces definiowania poszczególnych okien przebiegał sprawniej. Podczas projektowania interfejsu użytkownika skorzystano również z języka *css*, służącego do zdefiniowania jednolitego stylu wszystkich ekranów wyświetlanych przez system.

W kolejnych podrozdziałach omówiono interfejs użytkownika programu *TMT Analyzer* z położeniem nacisku na przedstawienie rozwiązań, które zapewniają spełnienie wymagań funkcjonalnych omówionych w rozdziale 3.1. W opisie tym przedstawiono również ogólny przepływ sterowania i przejść między poszczególnymi ekranami.

6.1 Główny ekran aplikacji

Na głównym ekranie interfejsu użytkownika umieszczono panel zawierający przyciski udostępniające wszystkie zaimplementowane funkcjonalności systemu (Rys. 6.1). Wyróżniono tutaj:

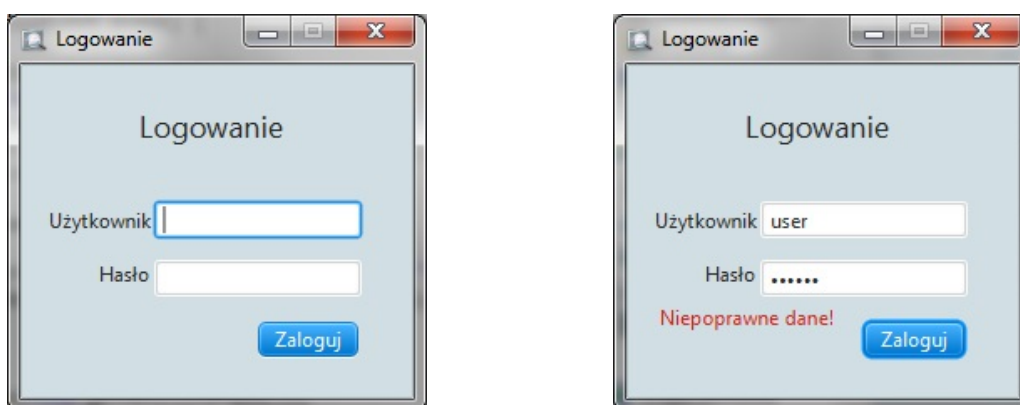
- Badanie - przeprowadzenie testu wraz z rejestracją jego przebiegu i wyznaczeniem podstawowych parametrów,
- Przegląd - dostęp do wyników badań i ich wizualizacji oraz do danych pacjentów znajdujących się w bazie,
- Analiza - przeprowadzenie szczegółowej analizy danych wraz z realizacją testów statystycznych mających na celu porównanie sparowanych populacji,
- Import - realizacja importu danych zawierających wyniki badań,
- Eksport - przeprowadzenie eksportu wybranych elementów lub całej zawartości bazy danych do zewnętrznego pliku,
- Opcje - dostęp do katalogów oraz pozostałej funkcjonalności systemu.



Rysunek 6.1. Główny ekran aplikacji.

6.2 Logowanie użytkownika

Logowanie jest pierwszą czynnością, jaką musi zrealizować użytkownik po uruchomieniu systemu *TMT Analzyzer*. Jest to element niezbędny do uzyskania dostępu do ekranu głównego aplikacji. Użytkownik powinien w określonych polach wprowadzić swoją unikalną nazwę, a także hasło (Rys. 6.2a). W przypadku podania nieprawidłowych danych logowania, ekran wyświetla adekwatny komunikat (Rys. 6.2b).



Rysunek 6.2. (a) ekran logowania, (b) błąd logowania.

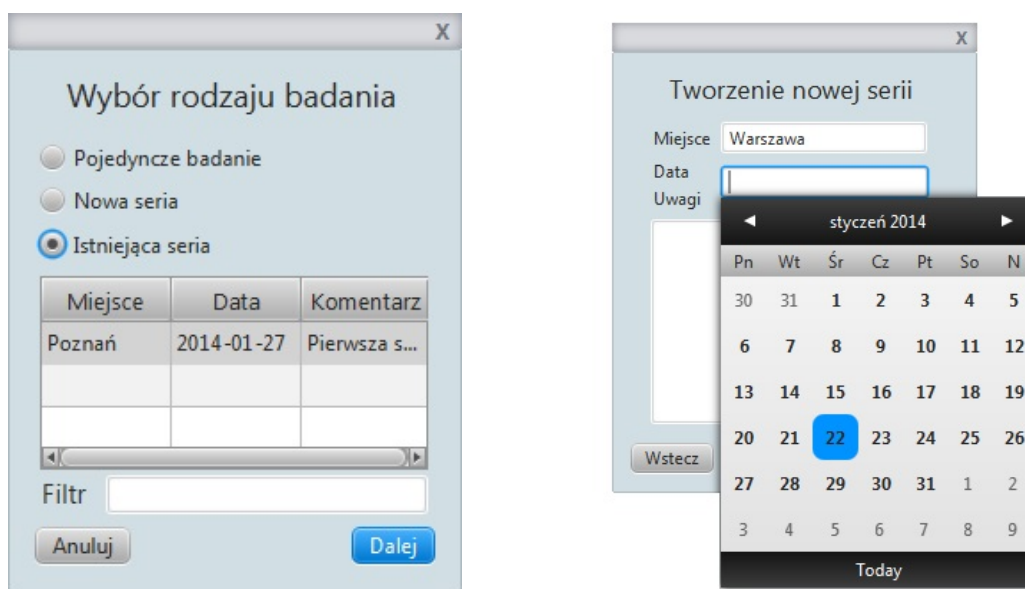
Celem określania aktualnego użytkownika jest udostępnienie funkcjonalności przypisanych mu przez rolę, jaką posiada w systemie. Wyróżniono role administratora oraz lekarza. Administrator posiada dostęp do wszystkich funkcji aplikacji, włącznie z dodawaniem, modyfikowaniem i usuwaniem użytkowników z systemu. Lekarzowi przypisano dostęp do modułów pozwalających mu na

przeprowadzanie testu, przegląd i analizę danych, ich import i eksport, a także dostęp do podstawowych katalogów.

6.3 Przeprowadzenie badania

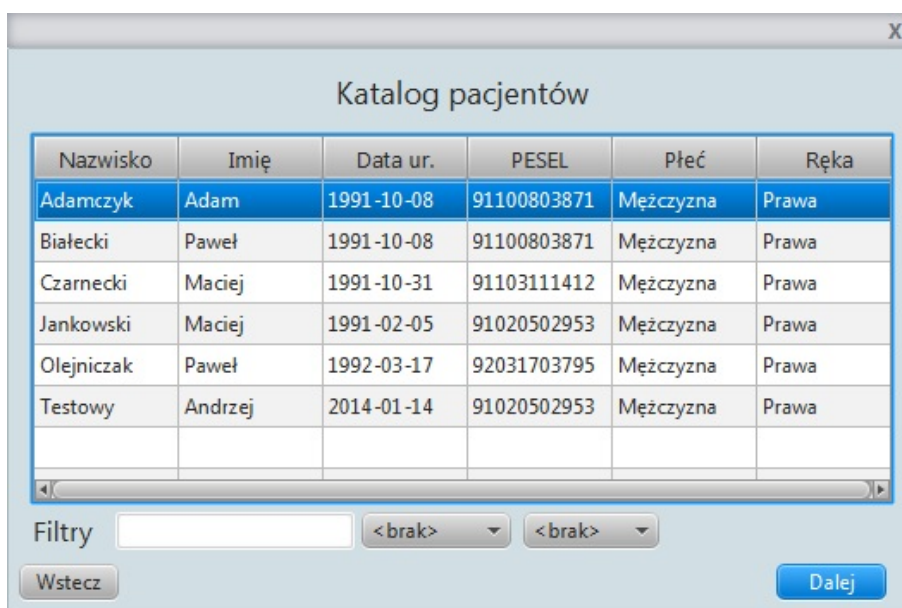
W celu uruchomienia modułu przeprowadzenia badania należy wybrać odpowiedni przycisk znajdujący się na górnym panelu przycisków umieszczonym w głównym oknie aplikacji (Rys. 6.1). Proces mający na celu zgromadzenie wszystkich niezbędnych danych dotyczących badania podzielono na kilka etapów.

Najpierw, system wyświetla okno, w którym użytkownik powinien określić czynność, którą chce wykonać: zrealizować pojedyncze badanie bez określania serii badań, zdefiniować nową lub wybrać jedną z istniejących serii badań, które dodatkowo można przefiltrować (Rys. 6.3a). Chcąc utworzyć nową serię badań, użytkownik powinien wprowadzić dane dotyczące jej lokalizacji oraz daty. Pole zawierające miejsce przeprowadzenia serii badań podlega walidacji i w przypadku jego pominięcia jest podświetlane w kolorze czerwonym. Wybór daty jest realizowany za pomocą przeznaczonego do tego wyskakującego okienka (Rys. 6.3b). Istnieje również opcja wprowadzenia uwag dotyczących serii badań, jednak nie są one wymagane przez system.



Rysunek 6.3. (a) Ekran wyboru rodzaju badania, (b) okno definiowania nowej serii badań.

Następnym krokiem jest określenie, czy dane dotyczące pacjenta poddawanego badaniu istnieją już w bazie czy też należy je do niej dodać. W przypadku konieczności dodania danych, system wyświetla okno, w którym użytkownik powinien podać podstawowe dane, umożliwiające jednoznaczную identyfikację nowego pacjenta. Chcąc przeprowadzić badanie na pacjencie, którego dane istnieją już w bazie danych, użytkownik powinien wybrać interesujący go rekord z tabeli zawierającej zdefiniowanych pacjentów. Tabela ta dodatkowo może podlegać filtracji (Rys. 6.4).



Nazwisko	Imię	Data ur.	PESEL	Płeć	Ręka
Adamczyk	Adam	1991-10-08	91100803871	Męczyzna	Prawa
Bialecki	Paweł	1991-10-08	91100803871	Męczyzna	Prawa
Czarnecki	Maciej	1991-10-31	91103111412	Męczyzna	Prawa
Jankowski	Maciej	1991-02-05	91020502953	Męczyzna	Prawa
Olejniczak	Paweł	1992-03-17	92031703795	Męczyzna	Prawa
Testowy	Andrzej	2014-01-14	91020502953	Męczyzna	Prawa

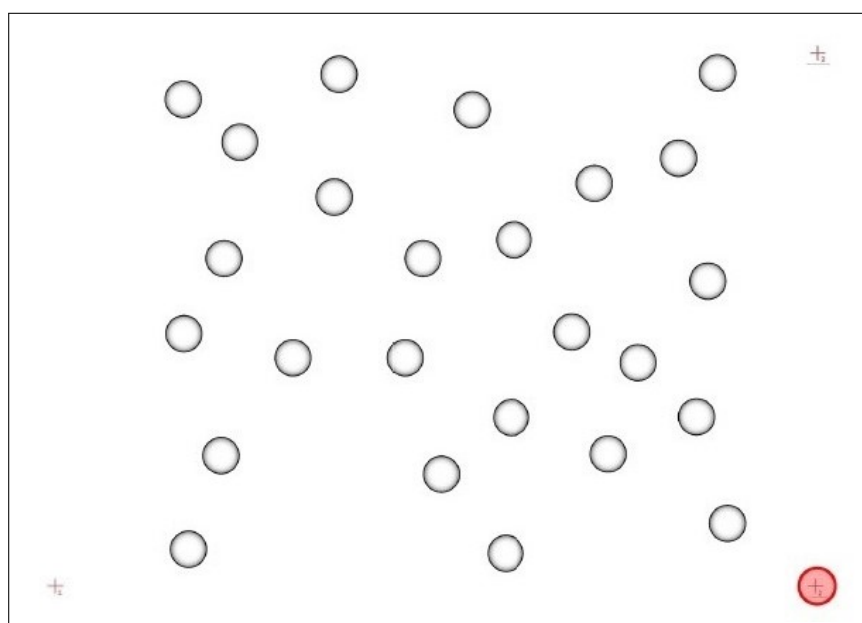
Filtry <brak> <brak>

Wstecz Dalej

Rysunek 6.4. Okno zawierające katalog zdefiniowanych pacjentów.

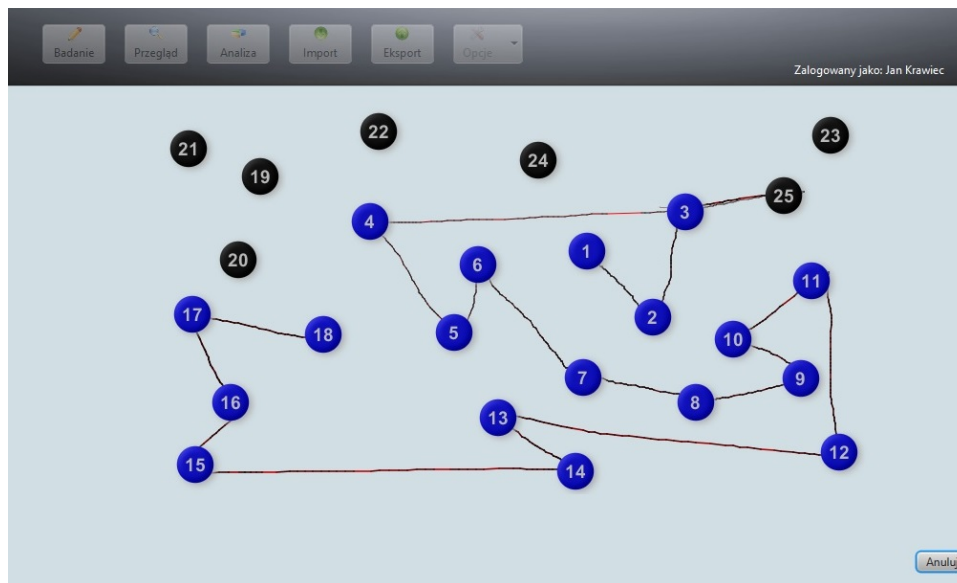
Następnie, system wyświetla okno, w którym użytkownik może zdefiniować dodatkowe informacje dotyczące przeprowadzanego badania, takie jak część realizowanego testu łączenia punktów. Ponadto, istnieje możliwość określenia zastosowanych środków medycznych oraz ich ilości mająca na celu dostarczenie szczegółowych danych dotyczących historii leczenia pacjenta.

Ostatnim etapem przygotowywania badania jest kalibracja tabletu, której szczegółowy opis umieszczono w rozdziale 5.2.3. W celu jej przeprowadzenia, system wyświetla serię trzech ekranów z wyraźnie zaznaczonymi punktami kalibracyjnymi, które użytkownik powinien wskazać w kolejności ich wyświetlania (Rys. 6.5).



Rysunek 6.5. Okno wyświetlane przez system, informujące o położeniu drugiego punktu kalibracyjnego.

Po wykonaniu powyższych operacji, system wyświetla okno z wyrysowanymi na nim punktami do połączenia, które odpowiadają punktom umieszczonym na arkuszu z testem zamontowanym na tablecie. Badanie rozpoczyna się w momencie wyjścia z punktu 1. Wówczas, wraz z ruchem ręki osoby badanej następuje jego wizualizacja na ekranie (Rys. 6.6).



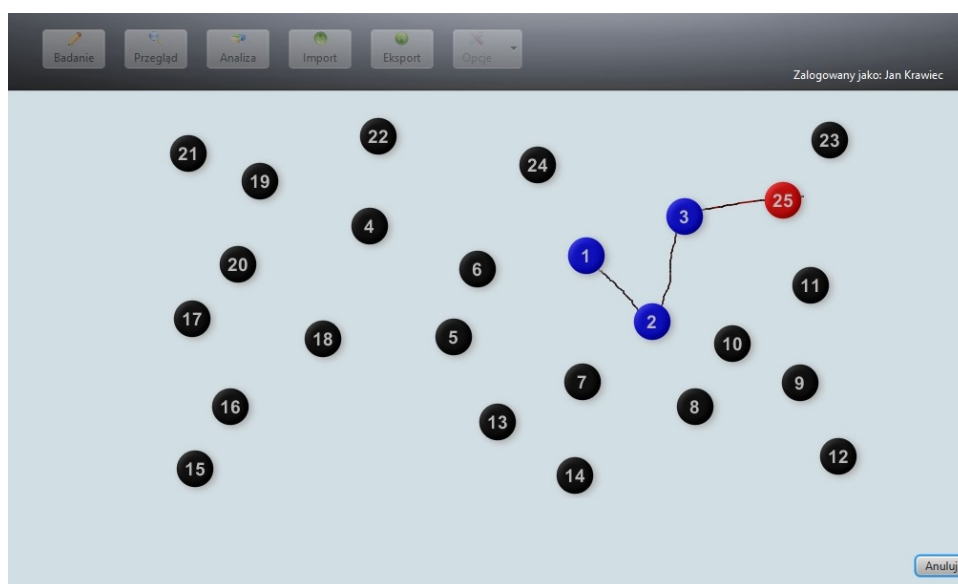
Rysunek 6.6. Przykładowy przebieg testu.

Podczas realizacji badania istnieje możliwość lokalizowania błędów pacjenta. Wówczas, punkty połączone w nieprawidłowej kolejności są oznaczane kolorem czerwonym, a zadaniem osoby kontrolującej badanie jest wycofanie pacjenta do ostatniego poprawnie połączonego punktu (Rys. 6.7). Po poprawieniu błędu pacjenta, czerwone oznaczenie znika z ekranu. Zakończenie badania następuje automatycznie po prawidłowym połączeniu wszystkich punktów. Wówczas użytkownik ma możliwość odrzucenia lub zapisania jego wyników, a także zrealizowania testu od nowa. Aby przerwać badanie w trakcie jego trwania, użytkownik powinien wybrać przycisk *Anuluj*, znajdujący się w dolnej części ekranu.

6.4 Przegląd wyników badań

System udostępnia moduł przeglądu zgromadzonych wyników badań. Celem tej funkcjonalności jest kontrolowanie procesu leczenia pacjentów, a także możliwość odtworzenia przebiegu danego testu na podstawie danych zgromadzonych na jego temat w bazie. Aby uzyskać dostęp do przeglądu badań, należy wybrać odpowiedni przycisk umieszczony na ekranie głównym aplikacji (Rys. 6.1).

Po uruchomieniu modułu przeglądu, system wyświetla ekran z tabelą zawierającą wszystkie dostępne badania, posortowane w zależności od dokładnej daty realizacji testu przez pacjenta. W tabeli tej umieszczono również dodatkowe informacje dotyczące badania, takie jak: część zrealizowanego testu, dane lekarza prowadzącego, rodzaj i wymiar zastosowanej kuracji medycznej. Uwzględniono tutaj również możliwość wyboru części przeprowadzonego testu łączenia punktów. Ponadto, umożliwiono przefiltrowanie wyników, mające na celu udostępnienie użytkownikowi wy-



Rysunek 6.7. Wyróżnienie błędnego przejścia między punktami.

łącznie interesujących go rekordów. Po wybraniu odpowiedniego badania, użytkownik powinien wcisnąć przycisk *Szczegóły*, znajdujący się w dolnej części ekranu.

Okno zawierające szczegóły dotyczące danego badania wyświetlane przez system podzielono na kilka sekcji. W pierwszej z nich, znajdującej się w górnej części ekranu, umieszczono dane personalne pacjenta i historię jego leczenia. Uwzględniono tutaj również dane lekarza prowadzącego oraz ewentualne komentarze dotyczące testu. System udostępnia możliwość analizy wyznaczonych parametrów statystycznych dla przejść między poszczególnymi punktami, a także ze względu na ich kierunek lub długość. W tym celu, w następnej sekcji okna przeglądu badań umieszczono przewijaną tabelę zawierającą wszystkie podstawowe parametry wyznaczone na podstawie zarejestrowanego przebiegu badania dotyczące zarówno pojedynczych przejść, jak i grup ruchów (Rys. 6.8).

Cecha	Całość	1	2	3
Czas trwania	42437.0	1217.0	886.0	2806.0
Czas rysowania	10197.0	672.0	299.0	434.0
Czas myślenia	32240.0	545.0	587.0	2372.0
Dystans	196.0	6.0	5.0	11.0
Liczba błędów	0.0	0.0	0.0	0.0
Liczba oderwań piórka	0.0	0.0	0.0	0.0
Maksymalna prędkość chwilowa	0.0626	0.0167	0.021	0.0549
Średnia prędkość chwilowa	0.0047	0.0066	0.0051	0.004
Odchylenie prędkości chwilowej	2.0E-4	5.0E-4	7.0E-4	7.0E-4
Maksymalne przyspieszenie	0.0042	8.0E-4	0.0015	0.0042
Średnie przyspieszenie	0.0094163222	4.2564E-6	7.3094E-6	4.6292
Odchylenie przyspieszenia	2.0E-4	0.0	0.0	0.0
Maksymalny nacisk	1.0	0.5894	0.651	1.0
Średni nacisk	2606.7205	0.5569	0.615	0.9096
Odchylenie nacisku	49.2272	0.0051	0.0023	0.0067

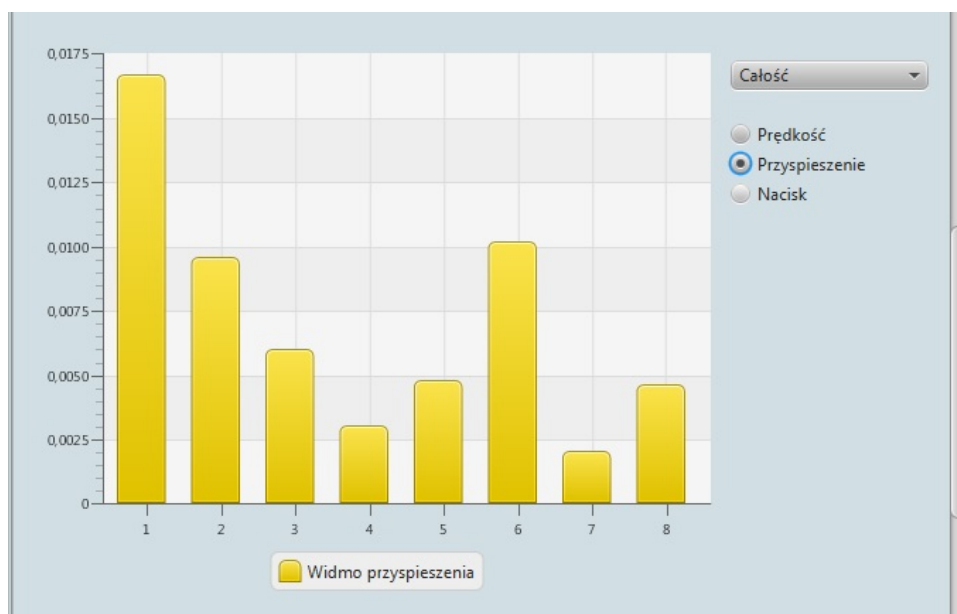
Poszczególne ruchy
 Grupy ruchów

Rysunek 6.8. Sekcja zawierająca szczegółowe dane dotyczące przebiegu badania.

Wyróżniono następujące grupy ruchów:

- ruchy w górę,
- ruchy w dół,
- ruchy w lewo,
- ruchy w prawo,
- długie przejścia,
- średnie przejścia,
- krótkie przejścia.

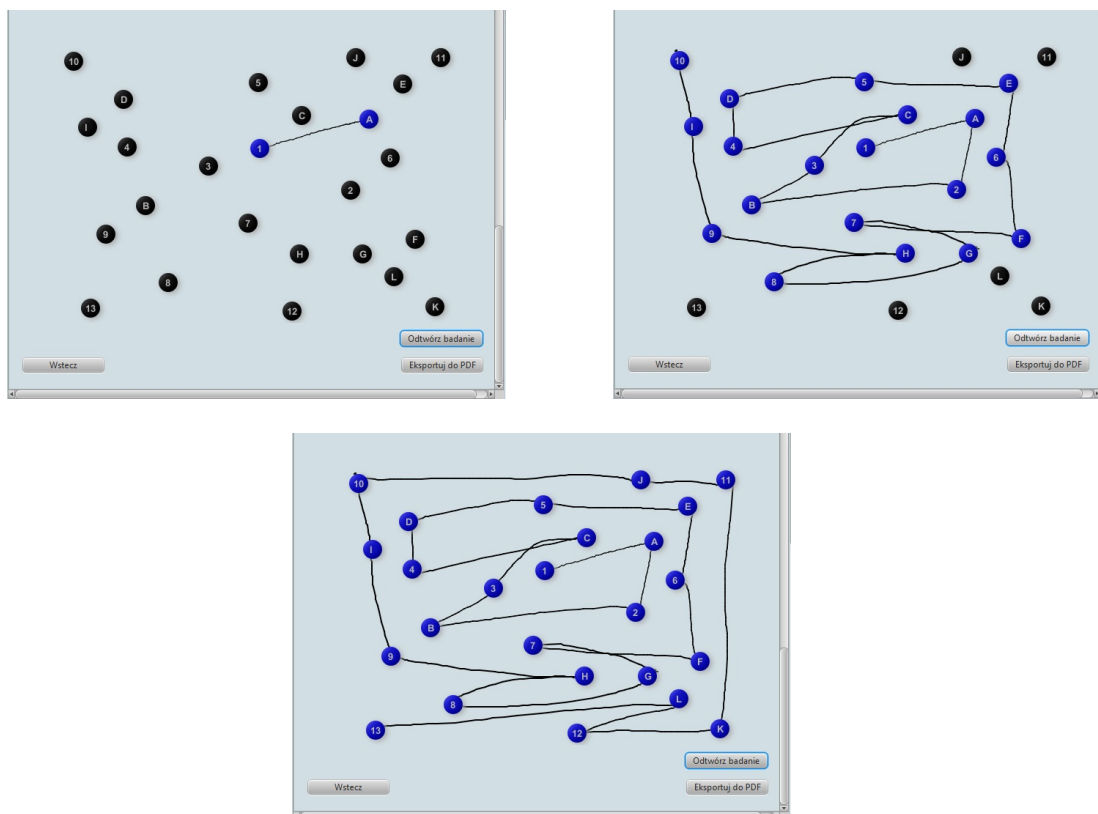
Poniżej umieszczono również sekcję odpowiedzialną za rysowanie widma drżeń określonej przez użytkownika cechy, której wybór odbywa się za pomocą przeznaczonych do tego listy z polami wyboru (ang. *radio buttons*) (Rys. 6.9). Istnieje również możliwość wybrania konkretnego przejścia między punktami w celu wyrysowania widma drżeń określonej cechy.



Rysunek 6.9. Wykres widma drżeń.

W ostatniej sekcji omawianego okna umieszczono wizualizację toru kreślenia (Rys. 6.10). Jej uruchomienie polega na wybraniu przycisku *Odtwórz*, znajdującego się ponad kanwą z wyrysowanymi punktami. Ich rozmieszczenie jest definiowane na podstawie części testu, której dotyczyło dane badanie. Po uruchomieniu wizualizacji, użytkownik posiada dostęp do dokładnego odwzorowania toru kreślenia, uwzględniając ewentualne błędy popełnione przez pacjenta. Warto zauważyć, że proces ten jest rozłożony w czasie, zgodnie z rzeczywistym czasem realizacji testu przez pacjenta.

System udostępnia również możliwość wygenerowania raportu w postaci pliku PDF. Zawiera on podstawowe dane osobowe pacjenta oraz szczegółowe informacje dotyczące przebiegu badania. Uwzględniono tutaj tabelę z wartościami parametrów statystycznych dotyczących poszczególnych przejść między punktami, a także wykresy przedstawiające widma drżeń prędkości, przyspieszenia i nacisku pióra na tablet. W celu utworzenia raportu, należy wybrać przycisk *Zapisz*, znajdujący się w dolnej części ekranu i w nowo wyświetlonym oknie podać ścieżkę do tworzonego pliku.



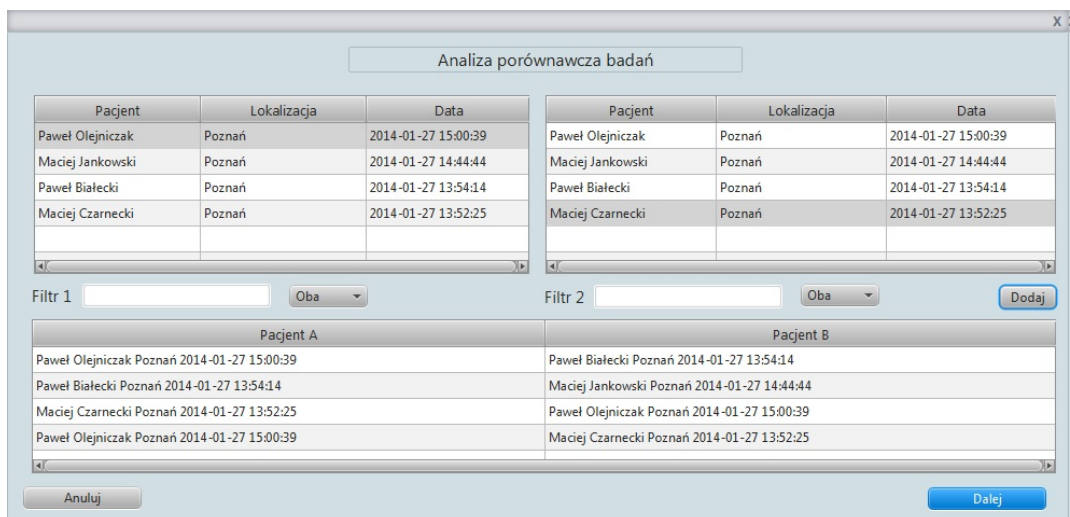
Rysunek 6.10. Przykładowe fazy odtwarzania toru kreślenia pacjenta.

6.5 Moduł analizy danych

System udostępnia możliwość wykonania analizy danych polegającej na przeprowadzeniu odpowiednich testów statystycznych, opierających się na sparowanych populacjach. Dodatkowo, zaimplementowano testowanie różnic w wynikach określonego pacjenta, obrazujących wpływ zastosowanej kuracji medycznej. Wówczas, użytkownik powinien połączyć badania danego pacjenta, poddawane weryfikacji. Istnieje również możliwość porównania wyników badań dwóch pacjentów. W celu uruchomienia modułu, należy wybrać odpowiedni przycisk znajdujący się na górnym panelu przycisków ekranu głównego aplikacji.

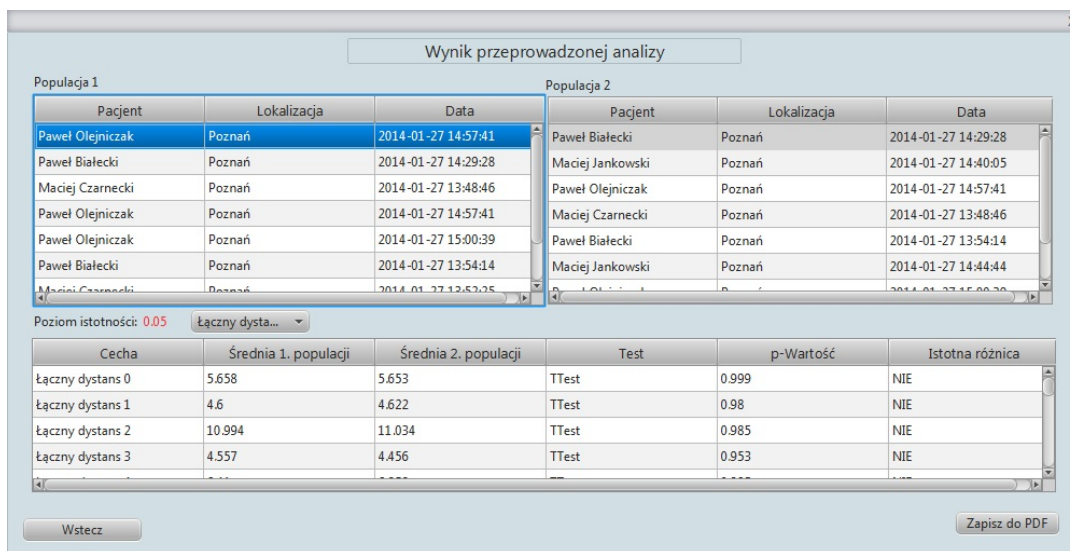
Pierwszym etapem analizy jest przeprowadzenie parowania pacjentów. System *TMT Analyzer* udostępnia ekran (Rys. 6.11) zawierający w swojej górnej części dwie tabele, w których wyszczególniono podstawowe informacje dotyczące badań mogących podlegać sparowaniu. Istnieje również możliwość filtrowania rekordów, a także grupowania badań ze względu na część przeprowadzonego testu łączenia punktów. Parowanie populacji polega na wskazaniu jednego rekordu z każdej tabeli, odpowiadającego badaniu, które użytkownik chciałby połączyć w parę. Następnie, wybranie przycisku *Dodaj* powoduje przeniesienie tej pary do tabeli znajdującej się w dolnej części ekranu. W taki sposób można sparować wiele rekordów, odpowiadających badaniom przeznaczonym do analizy. W celu sprawdzenia różnic w wynikach badań danego pacjenta, a także w wynikach dwóch różnych pacjentów, użytkownik powinien w lewej tabeli wskazać pierwsze badanie, a w prawej tabeli drugie badanie poddawane weryfikacji. Po ukończeniu tego procesu, należy wcisnąć przycisk *Dalej*.

Drugim etapem procesu analizy danych jest przeprowadzenie testów weryfikacji hipotez statystycznych. W zależności od liczebności sparowanych populacji, a także rozkładów ich cech, wyko-



Rysunek 6.11. Ekran umożliwiający parowanie populacji.

nywane są odpowiednie testy statystyczne. Ich szczegółowy opis umieszczono w rozdziale 5.3.5. Po ich ukończeniu, system wyświetla ekran zawierający wyniki przeprowadzonej analizy (Rys. 6.12). W jego górnej części umieszczono informacje dotyczące sparowanych populacji, natomiast w dolnej części znajduje się dokładny raport z przebiegu testów. Zawarto w nim zagregowane wartości określonej cechy obydwu populacji, rodzaj przeprowadzonego testu statystycznego, a także dokładną p-wartość i odpowiedź dotyczącą istotnej statystycznie różnicy zachodzącej między populacjami na danej cesze.



Rysunek 6.12. Raport zawierający wyniki przeprowadzonych testów statystycznych.

W przypadku weryfikacji różnic między wynikami badań pojedynczego pacjenta, po jej ukończeniu system wyświetla okno zawierające szczegółowe informacje dotyczące wartości parametrów statystycznych wynikających z poszczególnych przejść między punktami oraz wyróżnienie ich zmian następujących wskutek zastosowanej terapii (Rys. 6.13).

Paweł Białecki

PESEL	91100803871
Data ur.	1991-10-08
Płeć	Męczyzna
Ręka	Prawa
Lekarz	Administrator Systemu
Miejsce badania	Poznań
Data badania	2014-01-27 14:29:28.768
Data drugiego badania	2014-02-05 22:27:59.982
Typ badania	TMT_A & TMT_B
Rodzaj operacji	
Przyjęta substancja	Vicodin
Ilość substancji	10
Komentarz	

Łączny dystans...

Cecha	wartość 1	wartość 2	Różnica
Łączny dystans ruc...	7.755	5.771	1.984
Łączny dystans ruc...	3.989	4.551	-0.562
Łączny dystans ruc...	8.893	10.547	-1.654
Łączny dystans ruc...	4.338	3.977	0.361
Łączny dystans ruc...	5.095	6.145	-1.05
Łączny dystans ruc...	8.353	8.964	-0.612
Łączny dystans ruc...	2.998	6.539	-3.541
Łączny dystans ruc...	7.207	6.989	0.217
Łączny dystans ruc...	6.69	8.532	-1.842
Łączny dystans ruc...	3.861	5.052	-1.19

Powrót Zapisz do PDF

Rysunek 6.13. Raport zawierający informacje o różnicach w wynikach badań danego pacjenta.

Analogicznie, po ukończeniu weryfikacji różnic między wynikami dwóch różnych pacjentów, system wyświetla okno w którym umieszczono podstawowe informacje ich dotyczące oraz szczegółowe wyniki analizy. Zawierają one dane dotyczące poszczególnych parametrów statystycznych wynikających z zarejestrowanych przebiegów obydwu badań i różnice następujące między nimi. Istnieje również możliwość wyboru cechy do analizy za pomocą dedykowanej do tego listy rozwijanej (Rys. 6.14).

Wynik przeprowadzonej analizy

Populacja 1			Populacja 2		
Pacjent	Lokalizacja	Data	Pacjent	Lokalizacja	Data
Maciej Czarnecki	Poznań	2014-01-27 13:48:46	Paweł Białecki	Poznań	2014-01-27 14:29:28
Maciej Czarnecki	Poznań	2014-01-27 13:52:25	Paweł Białecki	Poznań	2014-01-27 13:54:14

Poziom istotności: 0.05 Łączny dystans...

Cecha	Wartość 1	Wartość 2	Różnica
Łączny dystans ruchu 0	5.412	5.771	-0.359
Łączny dystans ruchu 1	4.789	4.551	0.238
Łączny dystans ruchu 2	11.134	10.547	0.587
Łączny dystans ruchu 3	4.664	3.977	0.687
Łączny dystans ruchu 4	6.284	6.145	0.139

Wstecz Zapisz do PDF

Rysunek 6.14. Raport zawierający informacje o różnicach w wynikach dwóch pacjentów.

Warto dodać, że każde z powyżej opisanych okien udostępnia możliwość utworzenia raportu w postaci pliku PDF zawierającego wyniki szczegółowej analizy danych, w zależności od zasto-

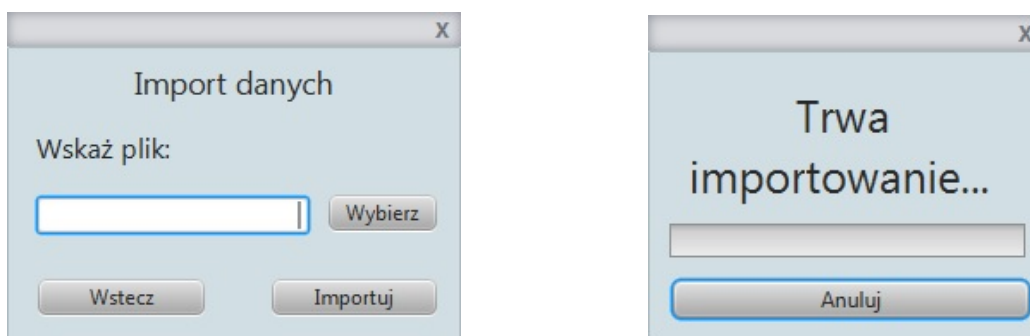
sowanej opcji porównywania badań. W tym celu, w oknie zawierającym rezultaty przeprowadzonych testów i weryfikacji należy wybrać odpowiedni przycisk, znajdujący się w dolnej części ekranu. Wówczas, system wyświetla okno umożliwiające określenie ścieżki do nowo tworzonego pliku. W każdym raporcie system umieszcza w postaci odpowiednich tabel wszystkie dane wyświetlane przez ekrany zawierające podsumowanie analizy. W taki sposób, informacje wygenerowane przez aplikację mogą zostać przechowane jako plik zewnętrzny lub wydrukowane.

6.6 Import i eksport danych

System udostępnia możliwość zaimportowania danych z zewnętrznego pliku mającego postać dokumentu *XML* a także eksportu danych (określonej części lub całości) do takiego pliku.

6.6.1 Import

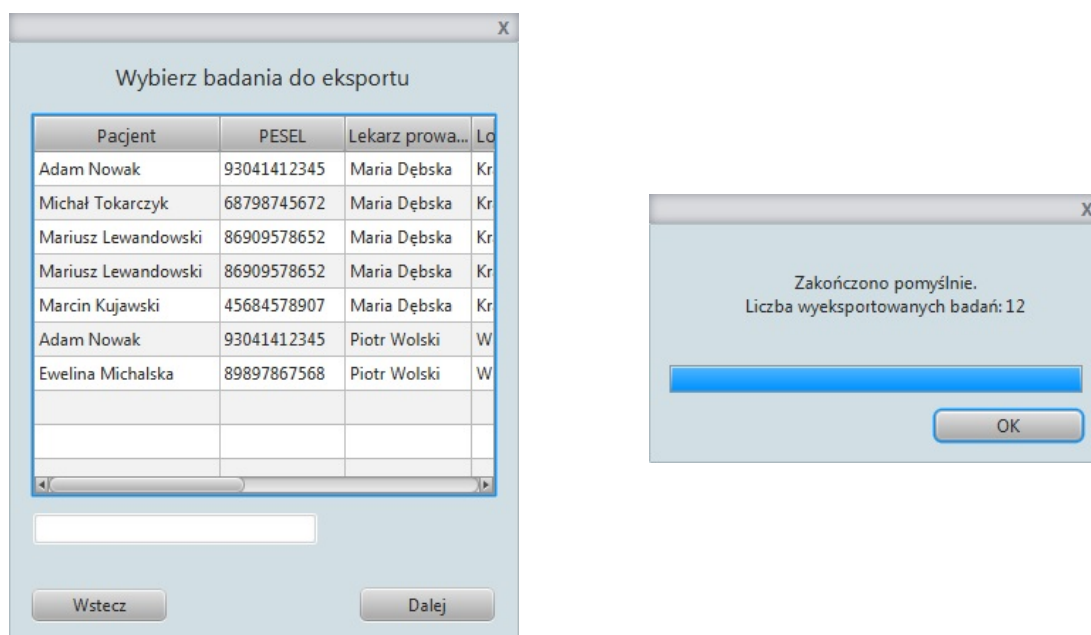
W celu skorzystania z funkcji importu danych do zewnętrznego pliku, użytkownik powinien wybrać odpowiedni przycisk z głównego ekranu aplikacji (Rys. 6.1). Wówczas, system wyświetla okno z polem umożliwiającym określenie pliku zewnętrznego (Rys. 6.15a). Podlega ono walidacji sprawdzającej, czy użytkownik wskazał plik w odpowiednim formacie. Po wciśnięciu przycisku *Importuj*, system wyświetla nowe okno zawierające pasek postępu realizacji procesu (Rys.6.15b). Po jego ukończeniu, użytkownik otrzymuje informację dotyczącą liczby zaimportowanych rekordów.



Rysunek 6.15. (a) pole wskazywania pliku do importu, (b) pasek postępu importu.

6.6.2 Eksport

Uzyskanie dostępu do opcji eksportu danych do zewnętrznego pliku polega na wybraniu odpowiedniego przycisku na górnym panelu głównego ekranu aplikacji (Rys. 6.1). Następnie, użytkownik powinien określić, czy chce przeprowadzić eksport wszystkich czy tylko wybranych badań. W przypadku wybrania tej drugiej opcji, system wyświetla okno z listą dostępnych badań (Rys. 6.16a). Zawarto w niej podstawowe dane pacjenta, lekarza prowadzącego oraz przeprowadzonego badania. Po wskazaniu przez użytkownika ścieżki do pliku zewnętrznego oraz określeniu jego nazwy, system wyświetla pasek postępu eksportu. Po pomyślnym zakończeniu realizacji procesu system wyświetla stosowny komunikat (Rys. 6.16b), natomiast użytkownik powinien za pomocą eksploratora systemu uzyskać dostęp do pliku *XML* zawierającego wyeksportowane dane.



Rysunek 6.16. (a) lista badań dostępnych do eksportu, (b) komunikat wyświetlany na zakończenie eksportu.

6.7 Pozostałe funkcjonalności systemu

TMT Analyzer udostępnia kilka przydatnych funkcjonalności, które wpływają na wygodę jego używania, jednak nie są kluczowe z punktu widzenia poprawnego działania systemu. Użytkownik może uzyskać do nich dostęp, wybierając przycisk *Opcje* znajdujący się na głównym ekranie aplikacji.

6.7.1 Dostęp do katalogów

System *TMT Analyzer* umożliwia dostęp do katalogów mających na celu udostępnianie użytkownikowi danych przechowywanych w bazie i wykorzystywanych przez niego podczas pracy, a także odpowiednią ich strukturalizację. Zaimplementowano również możliwość definiowania nowych rekordów oraz modyfikacji i usuwania istniejących, a także filtracji zawartości poszczególnych katalogów. Wyróżniono katalogi: pacjentów, pracowników, leków, chorób i operacji. Warto zauważyć, że dostęp do katalogu pracowników posiadają wyłącznie użytkownicy będący administratorami systemu.

6.7.2 Wylogowanie

Ostatnią dodatkową funkcją udostępnianą przez system jest wylogowanie użytkownika. Po wylogowaniu, system wyświetla ekran przedstawiony na Rysunku 6.2a. Jest ono również realizowane automatycznie podczas zamykania aplikacji.

Rozdział 7

Zakończenie

Cel przedsięwzięcia realizowanego w ramach projektu inżynierskiego, którym było opracowanie systemu *TMT Analyzer* wspomagającego przeprowadzanie i analizę tableтового testu łączenia punktów został osiągnięty. W zamierzeniu twórców, aplikacja znacząco wpłynie na podniesienie poziomu świadczonych przez użytkowników usług medycznych, a także na ich bardziej efektywną pracę. Automatyzacja procesu przeprowadzania badań pozwoli na sprawną i dokładną rejestrację ich przebiegu, umożliwiając dodatkowo weryfikację błędów osoby poddawanej testowi i wizualizację ruchów jej ręki. Dostęp do wyników przeprowadzonych badań oraz wizualizacja parametrów wyznaczonych na ich podstawie ułatwi stawianie trafnych diagnoz medycznych i podejmowanie właściwych decyzji, mających na celu poprawę stanu zdrowia danego pacjenta. Ponadto, poza możliwością przeprowadzenia analizy całościowej, pozwoli na szczegółową analizę zrealizowanego testu łączenia punktów (ang. *Trail Making Test*) ze względu na kierunki oraz długości poszczególnych przejść między punktami, mającą na celu weryfikację problematycznych ruchów. System umożliwia również wyznaczanie czasu ruchu ręki pacjenta, a także czasu jego zastanawiania się. Ważnym zadaniem modułu analizy danych jest weryfikowanie postępu w wynikach badań określonego pacjenta, a także umożliwienie użytkownikowi przeprowadzenia testów statystycznych badających istotne różnice między dwiema sparowanymi populacjami. System ponadto wpłynie na poprawienie struktury przechowywanych danych, ułatwiając ich obsługę i organizację. Umożliwiając realizację importu i eksportu danych, spowoduje łatwe przenoszenie i udostępnianie badań innym pracownikom. Dodatkowo, dzięki katalogom zawierającym dane dotyczące m.in. pacjentów, operacji lub leków, ułatwi użytkownikom pracę z systemem poprzez zapewnienie odpowiedniej strukturalizacji przydatnych im informacji.

Głównym zadaniem systemu *TMT Analyzer* jest pokonanie problemu niskiej jakości danych, wynikającego z zastosowania tradycyjnych metod ich pozyskiwania. Za pomocą zrealizowanego systemu, użytkownicy będą mieli dostęp do danych cyfrowych, które mogą podlegać wielokrotnej i szczegółowej analizie. Podczas realizacji testów, mających na celu weryfikację poprawności działania poszczególnych modułów aplikacji, uznano, że spełnia ona wszystkie zdefiniowane wymagania dotyczące funkcjonalności i udostępnia rozwiązania, które mogą znacznie wpłynąć na komfort i jakość pracy docelowych użytkowników. Jednak dopiero oni będą mogli dokonać rzeczywistej oceny systemu, a tym samym uzasadnienia jego realizacji.

Literatura

- [Wol09] Małgorzata Wolny, *Choroby społeczne i cywilizacyjne człowieka*, 2009.
- [Mit05] Maura Mitrushina, *Handbook of Normative Data for Neuropsychological Assesment*, Oxford University Press, 2005.
- [Tra14] sv.wikipedia:Trail Making Test, [on-line] [http : //sv.wikipedia.org/wiki/Trail_making_test](http://sv.wikipedia.org/wiki/Trail_making_test), 2014.
- [Jas97] Andrzej Jaskiewicz, *Inżynieria oprogramowania*, Helion, 1997.
- [Zak06] Maciej Zakrzewicz, *Projektowanie aplikacji J2EE w architekturze Model-View-Controller*, 2006.
- [Hib14] pl.wikipedia:Hibernate, [on-line] [http : //pl.wikipedia.org/wiki/Hibernate](http://pl.wikipedia.org/wiki/Hibernate), 2014.
- [Ora99] Oracle[®], *Code Conventions for the Java Programming Language*, [on-line] [http : //www.oracle.com/technetwork/java/codeconv - 138413.html](http://www.oracle.com/technetwork/java/codeconv-138413.html), 1999.
- [Hol05] Allen Holoub, *Wzorce projektowe - analiza kodu sposobem na ich poznanie*, Helion, 2005.

Dodatek A

Załączniki

1. Płyta CD-ROM

zawartość płyty:

- Folder "med_project" z kodem źródłowym projektu,
- Plik "Praca inżynierska" z treścią pracy dyplomowej.



© 2014 Paweł Białecki, Maciej Czarnecki, Maciej Jankowski, Paweł Olejniczak

Instytut Informatyki, Wydział Informatyki
Politechnika Poznańska

Skład przy użyciu systemu L^AT_EX.

Bib_TE_X:

```
@bachelorsthesis{ key,  
  author = "Paweł Białecki \and Maciej Czarnecki \and Maciej Jankowski \and Paweł Olejniczak",  
  title = "{System realizacji oraz analizy  
tabletowego testu łączenia punktów}",  
  school = "Poznan University of Technology",  
  address = "Pozna{\n}, Poland",  
  year = "2014",  
}
```