TEMAT PRACY:

ANALIZA MOŻLIWOŚCI WYKORZYSTANIA DARMOWYCH MAP DO UTWORZENIA PROSTEGO PROGRAMU DO ŚLEDZENIA POŁOŻENIA OBIEKTÓW WRAZ Z WIZUALIZACJĄ OBSZARU PRZESZUKANEGO

Spis treści

1. Wstęp

1.1. Cel Pracy

Celem niniejszej pracy jest opracowanie algorytmu umożliwiającego wizualizację obszaru przeszukanego przez bezzałogowe statki powietrzne, na podstawie zebranych danych geolokalizacyjnych obiektów oraz zaprojektowanie i implementacja systemu informatycznego, umożliwiającego określanie położeń geograficznych pojazdów w czasie rzeczywistym oraz ich wizualizację na urządzeniach mobilnych z systemem operacyjnym Android wraz z analizą możliwości wykorzystania API OpenStreetMap.

1.2. Wprowadzenie

Branża bezzałogowych statków latających, czy jak są one powszechnie nazywane dronów jest w tej chwili jedną z najbardziej przyszłościowych i najszybciej rozwijających się gałęzi rynku nowoczesnych technologii na świecie.

Pojazdy te przez długi czas były wykorzystywane głównie w przemyśle militarnym, jednak na przestrzeni ostatnich lat znajdują zastosowanie w coraz szerszym obszarze gałęzi usług i przemysłu sektora cywilnego. Są wykorzystywane chociażby przez służby ratunkowe do poszukiwania osób zaginionych lub ofiar katastrof, do patrolowania granic, monitorowania upraw rolniczych, obserwacji zwierząt na pastwiskach, nagrywania wydarzeń z perspektywy niedostępnej dla człowieka, kontroli bezpieczeństwa podczas imprez masowych, czy na dużych obszarach rekreacyjnych lub terenach zagrożonych pożarami. Są one doskonałym rozwiązaniem zawsze tam, gdzie trzeba przeprowadzić badania, bądź obserwacje w trudno dostępnych lub toksycznych dla człowieka warunkach np. w górach, czy gęstych lasach.

Jednoczesne wykorzystanie dronów wraz z zaawansowanymi systemami wizyjnymi oraz postępujący rozwój systemów informacji geograficznych GIS wraz z szeroką dostępnością systemów nawigacji satelitarnej oraz łatwością integracji z powszechnie stosowanymi urządzeniami mobilnymi stwarzają szerokie możliwości na tworzenie coraz to nowszych rozwiązań wspomagających wykorzystanie bezzałogowymi statków latających.

Niniejsza praca skupia się właśnie na analizie i opracowaniu jednego z takich rozwiązań, czyli systemu informatycznego służącego do kontroli położenia i wizualizacji obszaru zarejestro-

wanego przez kamery zamontowane w dronach, na podstawie algorytmu uwzględniającego geolokalizację statków. Na system ten składają się aplikacja mobilna do zbierania danych satelitarnych, pełniąca rolę geolokalizatora do potencjalnego zamontowania na dronie, część serwerowa komunikująca się z innymi elementami systemu i zajmująca się analizą zebranych danych, wraz z wyznaczaniem obszaru przeszukanego na podstawie opracowanego w trakcie tworzenia tej pracy algorytmu oraz mobilna aplikacja kliencka służąca do ich wizualizacji.

Zastosowanie projektowanego systemu jest wysoce uniwersalne i możliwe do dalszego rozszerzenia poprzez integrację z dodatkowymi czujnikami, czy systemami wizyjnymi. Na potrzeby tej pracy został on opracowany z myślą o zadaniach związanych z nadzorem i kontrolą dużych obszarów terytorialnych przez bezzałogowe statki powietrzne.

2. Dziedzina problemu

2.1. Najważniejsze pojęcia

W tej części pracy zostaną przedstawione najważniejsze pojęcia związane z teoretycznym aspektem pracy. W dalszych podrozdziałach wymienione pojęcia zostaną omówione w bardziej szczegółowy sposób.

Bezzałogowy statek powietrzny BSP - (UAV – Unmanned Aircraft Vehicle) - potocznie dron. Jest to pojazd, który nie wymaga do lotu załogi obecnej na pokładzie oraz nie ma możliwości zabierania pasażerów, pilotowany zdalnie lub wykonujący lot autonomicznie na podstawie zaprogramowanej trasy.

Dane geolokalizacyjne - dane określające położenie geograficzne obiektów w przestrzeni w stosunku do przyjętego układu odniesienia.

System wizyjny - układ współpracujących ze sobą elektronicznych urządzeń, którego funkcją jest automatyczna analiza wizyjna otoczenia na podobieństwo zmysłu wzroku u ludzi.

Obszar przeszukany - reprezentacja obszaru zarejestrowanego przez kamerę zamontowaną w dronie.

Numeryczny model terenu NMT - numeryczna reprezentacja powierzchni ziemskiej

SRTM - Shuttle Radar Topography Mission - międzynarodowa misja kosmiczna, mającą na celu zebranie najbardziej kompleksowego i dokładnego numerycznego modelu terenu (NMT) Ziemi.

Geoida - teoretyczna powierzchnia ekwipotencjalna, pokrywająca się w przybliżeniu z powierzchnią oceanów przy pełnej równowadze znajdujących się w nich mas wody.

Elipsoida ziemska - spłaszczona elipsoida obrotowa, której powierzchnia jest najbardziej zbliżona do hydrostatycznej powierzchni Ziemi.

Wysokość ortometryczna - odległość mierzona w stosunku do geoidy, wzdłuż linii pionu w rzeczywistym polu siły ciężkości, utożsamiana z wysokością nad poziomem morza.

Wysokość elipsoidalna - odległość mierzona w stosunku do elipsoidy

Undulacja - różnica wysokości między geoidą, a elipsoidą

WGS84 - model powierzchni ziemskiej opracowany w stosunku do elipsoidy WGS84,

EGM96 - model powierzchni ziemskiej opracowany w stosunku do geody EGM96,

2.2. Geodezyjne powierzchnie odniesienia

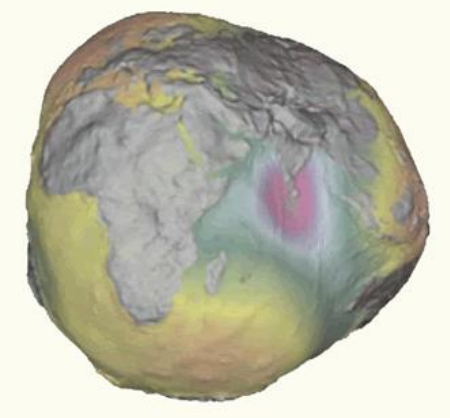
Powszechnie wiadomo, że rzeczywisty kształt powierzchni ziemskiej jest bardzo nieregularny i praktycznie niemożliwy do opisania matematycznego. Wpływa na niego wiele czynników takich jak: ruch obrotowy i obiegowy planety, cieplny i grawitacyjny wpływ ciał niebieskich, własności fizyczne litosfery i hydrosfery i wiele innych. Systemy nawigacyjne, aby być w stanie określić położenie obiektów w stosunku do powierzchni ziemskiej muszą znać jej rzeczywisty kształt i rozmiar, a przynajmniej takie ich przybliżenie, które zapewni wymaganą dokładność wykonywania pomiarów i obliczeń nawigacyjnych. W związku z tym w rzeczywistości wykorzystuje się modele

Ziemi, które w zależności od skali i zakładanej dokładności za powierzchnię odniesienia przyjmują geoidę lub elipsoidę.

2.2.1. Geoida

Jeśli wyobrazimy sobie, że powierzchnia ziemska jest w pełni pokryta wodą oraz zignorujemy wpływy falowań oraz prądów morskich, na cząsteczki wody powstałego w ten sposób "globalnego oceanu" będzie oddziaływać jedynie siła grawitacji. Otrzymana w ten sposób powierzchnia nie będzie jednak regularna ponieważ jej kształt będzie zależny od nieregularnego rozmieszczenia masy wewnątrz planety, wpływającego na kierunki działania siły grawitacji dla każdego punktu powierzchni. W rzeczywistości na nieregularność poziomu wód ma wpływ więcej czynników takich jak temperatura wody, jej zasolenie, czy wpływ grawitacyjny innych obiektów takich jak góry. Geoida jest zatem teoretyczną powierzchnią ekwipotencjalną, pokrywającą się w przybliżeniu z powierzchnią oceanów przy pełnej równowadze znajdujących się w nich mas wody i jest w każdym swym punkcie prostopadła do kierunku siły ciężkości.

Ze względów praktycznych geoida często jest utożsamiana ze średnim poziomem morza, chociaż w rzeczywistości w niektórych miejscach może odbiegać od niego nawet o kilka metrów.



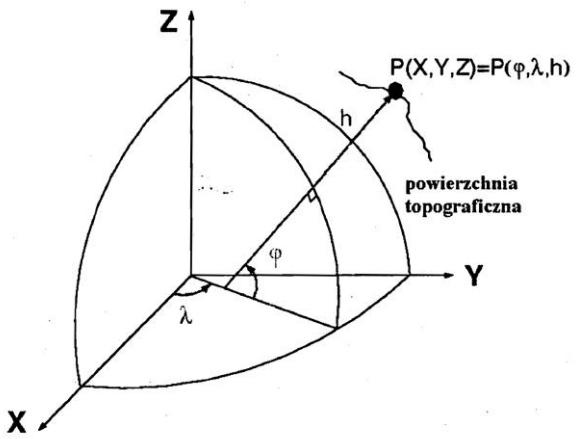
Rys. 1 W wyolbrzymiony sposób zdeformowana powierzchnia geoidy, pokazująca stopień jej skomplikowania

2.2.2. Elipsoida

Elipsoida jest powierzchnią powstałą na skutek obrotu elipsy wokół jej osi symetrii. W przypadku Ziemi osią tą jest mała oś elipsy, odpowiadająca osi ziemskiej. Elipsoidą ziemską nazywamy spłaszczoną elipsoidę obrotową, której powierzchnia jest najbardziej zbliżona do hydrosta-

tycznej powierzchni Ziemi. Współcześnie parametry elipsoidy wyznaczane są na podstawie pomiarów satelitarnych.

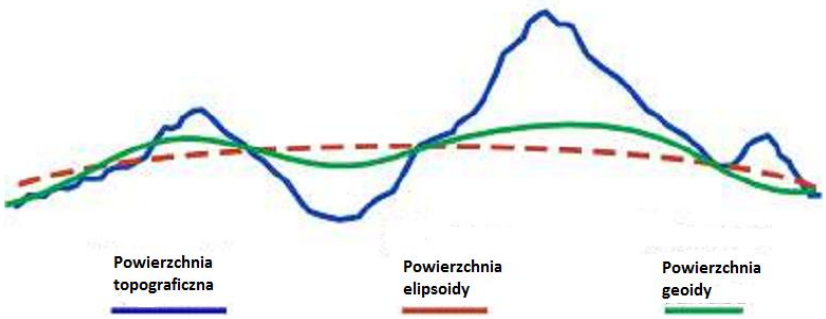
Z elipsoidą ziemską poza układem współrzędnych kartezjańskich, wiąże się układ współrzędnych geograficznych (elipsoidalnych). W układzie tym występują trzy współrzędne: szerokość geograficzna, długość geograficzna oraz wysokość elipsoidalna. Szerokością geograficzną nazywamy kąt φ zawarty między normalną do elipsoidy w punkcie obserwatora, a płaszczyzną równika. Długością geograficzną nazywamy kąt dwuścienny λ pomiędzy płaszczyzną południka zerowego i płaszczyzną południka obserwatora. Wysokość elipsoidalna h to odległość euklidesowa pomiędzy punktem obserwatora, a jego rzutem wzdłuż normalnej na elipsoidzie.



Rys. 2 Współrzędne geograficzne (elipsoidalne)

Powiązanie układu współrzędnych geograficznych z punktem związanym z Ziemią daje nam ziemski układ odniesienia. Natomiast powiązanie ze środkiem mas Ziemi i nadanie wartości liczbowych poszczególnym parametrom elipsoidy (oraz określenie pewnym stałych geofizycznych) daje nam geocentryczny układ odniesienia.

Powierzchnie geoidy oraz elipsoidy znacznie się różnią (Rys.3). Odstępy między geoidą, a elipsoidą nazywamy undulacjami geoidy, a ich wartości wahają się od -110m do +84m.



Rys. 3 Porównanie powierzchni

2.2.3. Porównanie modeli WGS84 oraz EGM96

Biorąc pod uwagę wysoce nieregularny kształt Ziemi praktycznie niemożliwym jest stworzenie modelu jednolicie dokładnego dla całej planety. W związku z tym przy dokładnych pomiarach i badaniach często wykorzystuje się modele opracowane lokalnie. Jednak popularyzacja sytemu nawigacji satelitarnej oraz konieczność współpracy międzynarodowej przy badaniach wymagały stworzenia modelu, który mógłby być wykorzystywany na całej planecie.

Grawitacyjny model EGM96 w bardzo dokładny sposób definiuje globalną geoidę ziemską, w ogólności tożsamą ze średnim poziomem morza, podczas gdy WGS84 przybliża powierzchnię ziemską jako geocentryczną elipsoidę, generalizującą kształt tej geoidy. EGM96 jest modelem znacznie bardziej złożonym i dokładnym, opartym na szczegółowej analizie ziemskiej siły grawitacyjnej, jednak jego użycie wymaga znacznie większych nakładów obliczeniowych. WGS84 określa elipsoidę najbardziej pasującą do geoidy EGM96. Model ten został utworzony w 1984 roku, jednak na przestrzeni lat był aktualizowany wraz z coraz to dokładniejszymi pomiarami geoidy. WGS84 nie jest modelem przestarzałym, jest po prostu znacznie uproszczonym modelem matematycznym, który mimo mniejszej dokładności wciąż jest często wykorzystywany ze względu na łatwość wykorzystania w analizie matematycznej. Model ten dla większości zastosowań jest wystarczająco dokładny. Jego niedokładność osiąga maksymalnie wartość 110m ze średnim błędem na poziomie 20-30m. Istnieje możliwość zredukowania tego błędu poprzez wprowadzenie korekty na podstawie znajomości undulacji geoidy na interesującym nas regionie.

2.3. Numeryczny Model Terenu NMT

2.3.1. Ogólnie o NMT

Numeryczny model rzeźby terenu (NMT) jest numeryczną reprezentacją powierzchni ziemskiej, utworzoną zazwyczaj przez zbiór punktów tej powierzchni oraz algorytmy, służące do aproksymacji jej położenia i kształtu na podstawie współrzędnych x, y, z tych punktów. Zawiera infor-

macje na temat wysokości poszczególnych punktów powierzchni terenu ponad ustalonym poziomem odniesienia np. nad poziomem morza.

Numeryczny model terenu można opracować na podstawie dowolnego zestawu danych mających charakter pola skalarnego, tzn. zawierających informacje o położeniu (współrzędne x i y w układzie współrzędnych) oraz wielkości skalara (współrzędna z).

Skonstruowanie prawidłowego modelu rzeźby terenu jest jednym z kluczowych wyzwań współczesnej geomatyki. Znajomość Numerycznego Modelu Terenu o wysokiej dokładności jest bardzo ważna z punktu widzenia wielu zastosowań praktycznych. Modele wysokości, w formie map topograficznych umożliwiają projektowanie przebiegu tras komunikacyjnych, wyznaczanie optymalnych lokalizacji masztów telefonii komórkowej, czy też wyznaczanie stref zalewowych. Dane reprezentujące NMT są niezbędne do prawidłowego funkcjonowania powietrznych systemów nawigacyjnych oraz powszechnie wykorzystywane w licznych Systemach Informacji Geograficznej.

Dane do opracowania NMT mogą być pozyskiwane na wiele sposobów takich jak: bezpośrednie pomiary terenowe, zdjęcia lotnicze i ich opracowania fotogrametryczne, mapy topograficzne, interferometria satelitarna, czy lotniczy skaning laserowy. Najczęściej stosowaną techniką pozyskiwania danych jest technologia fotogrametryczna. Na podstawie odpowiednich zdjęć lotniczych (tzw. fotogramów) budowany jest model stereoskopowy, na którym wykonuje się następnie stereodigitalizację powierzchni terenu, co w efekcie umożliwia odtwarzanie kształtów, rozmiarów i wzajemnego położenia obiektów w terenie. Tworzenie NMT na obszarach silnie zalesionych, bądź zurbanizowanych jest możliwe za pomocą technologii kartograficznej, polegającej na wektoryzacji warstwic na podstawie zeskanowanych diapozytywów map topograficznych. Do szybkiej budowy NMT o dużej dokładności jest stosowany skaning laserowy, który jest niezależny od warunków oświetlenia i pozwala na uzyskanie precyzyjnego modelu rzeźby terenu także dla obszarów o zwartej pokrywie roślinnej, czy w terenach zabudowy miejskiej.

Wymienione wyżej sposoby gromadzenia danych do modelowania rzeźby terenu łączy jedna wspólna cecha. Mają naturę lokalną tzn. wymagają w mniejszym lub większym stopniu bezpośredniej eksploracji badanego terenu, co jest bardzo kosztowne i czasochłonne oraz często niemożliwe z powodów politycznej niedostępności niektórych regionów świata. W efekcie przez wiele lat nie istniał jednolity dla całej ziemi numeryczny model terenu. Większość rozwiniętych państw tworzyło swoje własne dane kartograficzne, które znacznie różniły się między sobą pod względem skal, rozdzielczości oraz punktów odniesienia i co za tym idzie były wysoce niespójne. Co więcej globalny obszar pokrycia był bardzo niejednolity. Wiele części świata takich jak Ameryka Południowa, czy Afryka cechowało się brakiem danych topograficznych o wysokiej dokładności.

Uzyskanie międzynarodowych, cyfrowych danych wysokościowych o spójnej skali i rozdzielczości poprzez konwencjonalne sposoby okazało się praktycznie niemożliwe, a jedynym sposobem na ich ujednolicenie było zastosowanie globalnie jednolitej techniki pozyskiwania danych. Taką techniką okazała się interferometria satelitarna wykorzystana podczas misji SRTM.

2.3.2. Misja SRTM

SRTM - The Shuttle Radar Topography Mission - jest międzynarodową misją kosmiczną mającą na celu zebranie najbardziej kompleksowego i dokładnego numerycznego modelu terenu (NMT) Ziemi.

Projekt został przeprowadzony w lutym 2000 roku podczas lotu promu kosmicznego Endeavour wspólnie przez trzy agencje kosmiczne: Narodową Agencję Aeronautyki i Przestrzeni Kosmicznej Stanów Zjednoczonych NASA, Niemiecką Agencję Kosmiczną DRL oraz Włoską Agencję Kosmiczną ASI.

Główną ideą misji było zapewnienie jednolitego pod względem dokładności i skali modelu topograficznego o rozdzielczości terenowej 1" dla wszystkich obszarów lądowych znajdujących się pomiędzy równoleżnikami 60^{o} szerokości geograficznej północnej, a 56^{o} szerokości geograficznej południowej, stanowiących około 80% powierzchni lądowej Ziemi. Jednym z założeń misji był maksymalny błąd wysokości, jakim miał sie charakteryzować NMT, który został ustalony na +/- 16m dla wysokości bezwzględnej oraz +/- 10m dla wysokości względnej na poziomie ufności równemu 90%.

W trakcie trwania całej misji zebrano ponad 12 terabajtów danych, które następnie zostały poddane dokładnej analizie. W sierpniu 2001 roku ogłoszono zakończenie misji SRTM, a dane wynikowe zostały umieszczone w Internecie.

Zgodnie z oficjalną specyfikacją misji dokładność pionowa danych wysokościowych SRTM-1 na obszarze Europy wynosi 6,2m dla wysokości bezwzględnej oraz 8,7m dla wysokości względnej. Błąd geolokalizacji jest na poziomie 8,8m. Wartości te są różne w zależności od kontynentu, ale wszędzie mieszczą się w założonej dokładności +/- 16m.

Wadą interferometrii satelitarnej do pozyskiwania modelu NMT jest uwzględnianie pokrywy roślinnej i zabudowy przy określaniu wysokości, co znacznie zmniejsza dokładność danych na tego typu obszarach.

Numeryczne modele SRTM są bezpłatnie dostępne w Internecie w formie plików reprezentujących rastrowy numeryczny model terenu o zadanej rozdzielczości na poziomie 1", 3" lub 30" w zależności od wersji danych. Modele te są szeroko wykorzystywane do badań naukowych takich jak

badania potencjału energii wiatrowej, analiza geomorfometryczna, czy analiz hydrologicznych. Są również podstawowym źródłem danych wysokościowych dla wielu aplikacji takich jak Google Earth, Nasa World Wind, Google Maps, Yahoo Maps, czy OpenStreetMap.

Wysokości w modelu SRTM-C odniesione są do geoidy EGM-96, natomiast położenia pikseli są przedstawione względem elipsoidy WGS 84.

2.4. System GPS

2.4.1. Ogólny opis

Skrót GPS pochodzi od angielskich słów Global Positioning System, czyli Światowy System Określania Położenia. Jest to amerykański system nawigacji satelitarnej przeznaczony do szybkiego i dokładnego wyznaczania współrzędnych geograficznych, określających pozycję anteny odbiornika w przestrzeni, obejmujący zasięgiem całą kulę ziemską.

2.4.2. Zasada działania

Na każdej z sześciu orbit konstelacji znajdują się cztery satelity na wysokości ponad 20000km nad powierzchnią Ziemi. Orbity są rozmieszczone wokół całej Ziemi i nachylone do powierzchni równika pod kątem 55^{o}. Każdy satelita transmituje dwa rodzaje sygnałów: L1(1575,42 MHz) i L2 (1227,60MHz). Sygnał L1 jest przetwarzany dwoma pseudo-przypadkowymi sygnałami zagłuszającymi: chronionym kodem P i kodem C/A (Course Aquisition - pseudolosowa sekwencja bitów, powtarzająca się cyklicznie co 1ms). Sygnał L2 zawiera jedynie kod P. Każdy satelita wysyła inny sygnał, co ułatwia odbiornikom rozpoznanie, z którego satelity pochodzi dany sygnał.

Zasada działania systemu opiera się na pomiarze odległości pomiędzy satelitą, który porusza się po ściśle wyznaczonej orbicie, a odbiornikiem. Znana odległość od satelity lokuje odbiornik na sferze o promieniu równym zmierzonej odległości. Znana odległość od dwóch satelitów lokuje odbiornik na okręgu będącym przecięciem dwóch sfer. Kiedy odbiornik zmierzy odległość od trzech satelitów, istnieją tylko dwa punkty, w których może się znajdować. Jeden z tych punktów można wykluczyć jako znajdujący się zbyt wysoko lub poruszający się zbyt szybko i w ten sposób wyznaczyć swoją pozycję. Aby poznać odległość od satelitów emitujących bardzo słabe sygnały (o mocy zbliżonej do szumu tła) dokonuje się pomiaru opóźnienia sygnału odebranego z poszczególnych satelitów. Odbiornik GPS dysponuje jednak tylko własnym zegarem kwarcowym. Wyznaczanie

godziny z dokładnością do nanosekund odbywa się poprzez odbieranie sygnału nie od trzech, a od czterech satelitów. Można wówczas wyliczyć zarówno rzeczywisty czas, jak i położenie.

2.4.3. Dokładność

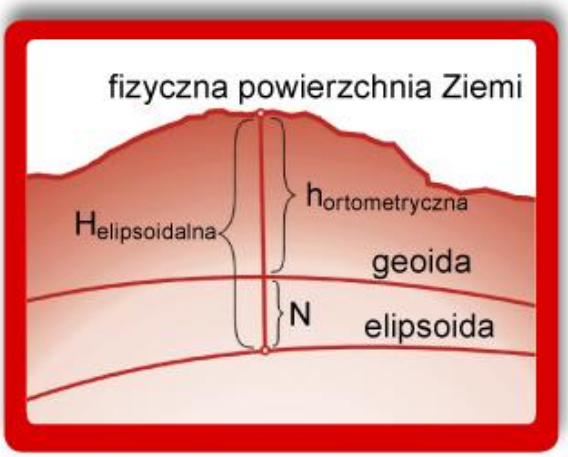
Dla typowych zastosowań dokładność pozycji wynosi do 10m. W przypadku dodatkowych wymogów dokładnościowych stosuje się metody zmniejszające błąd pozycjonowania takie jak metody uśredniające lub pomiar względny DGPS. Można wówczas uzyskać dokładność na poziomie kilku metrów.

Błąd wyznaczania wysokości jest zazwyczaj 1,5 razy większy od błędu położenia poziomego. Wynika to z geometrii satelitów, z których korzysta odbiornik. W celu uzyskania jak najbardziej dokładnej wysokości, należy używać satelitów zlokalizowanych jak najdalej od siebie, ale niezbyt nisko nad horyzontem oraz jednego dokładnie nad głową. Z reguły jednak odbiornik częściej wybiera satelity bliższe linii horyzontu w celu uzyskania bardziej dokładnej pozycji poziomej, na której zależy większości użytkowników odbiorników GPS. Powoduje to duże różnice wartości wysokości w danym punkcie.

2.4.4. Określanie wysokości

W wyniku pomiarów GPS otrzymujemy przestrzenne współrzędne prostokątne X, Y, Z pozycji anten odbiorników, które mogą następnie zostać przeliczone na współrzędne geodezyjne B, L, H odniesione do układu WGS84, czyli do elipsoidy WGS84, gdzie B i L to długość i szerokość geograficzna, a H to wysokość elipsoidalna.

Jednak rzadko kiedy wysokość elipsoidalna jest wysokością docelową, którą chcemy otrzymać. Zazwyczaj interesuje nas wysokość w odniesieniu do poziomu morza, który jak zostało opisane w podrozdziale "Geodezyjne powierzchnie odniesienia" w przybliżeniu odpowiada geoidzie.



Rys. 4 Pomiar wysokości

Wysokość liczona od poziomu morza jest określana wysokością ortometryczną i otrzymujemy ją odejmując od wysokości elipsoidalnej, otrzymanej z pomiarów GPS, wysokość geoidy N, czyli odstępu między geoidą, a elipsoidą. Wartości odstępów geoidy od elipsoidy są nazywane undulacjami geoidy i wahają się od -110m do +84m. W Europie elipsoida WGS84 jest średnio o około 30m wyżej niż rzeczywisty poziom morza. Część nowoczesnych odbiorników GPS posiada wbudowaną możliwość konwersji wysokości, jednak większość dostępnych urządzeń na rynku nie posiada takiej opcji. W celu określenia wysokości nad poziomem morza należy wówczas wprowadzić własne poprawki na podstawie znajomości lokalnych undulacji geoidy.

Alternatywą dla pomiaru wysokości z wykorzystaniem systemu GPS są wysokościomierze barometryczne, określające wysokość na podstawie zależności między zmianą ciśnienia atmosferycznego, a zmianą wysokości. Jednak ze względu na założenie wykorzystania w pracy urządzenia z systemem Android oraz fakt, iż większość takich urządzeń nie posiada czujników barometrycznych, w pracy zostanie wykorzystany pomiar wysokości przy wykorzystaniu systemu nawigacji satelitarnej.

3. Technologie dostępne na rynku

W tym rozdziale zostanie dokonany opis i porównanie dostępnych na rynku technologii związanych z głównymi aspektami projektowanego systemu oraz zostaną wybrane te, które nadają się najlepiej dla danego rozwiązania.

3.1. Wybór rodzaju aplikacji klienckiej

Podchodząc do procesu projektowania systemu informatycznego jedną z podstawowych kwestii jaką należy rozważyć jest urządzenie docelowe, na którym ma funkcjonować oprogramowanie. Wybór ten zależy w głównej mierze od przeznaczenia systemu i grupy docelowej jego użytkowników.

3.1.1. Aplikacja desktopowa

Aplikacja desktopowa tworzona jest dla użytkowników laptopów i komputerów stacjonarnych. Wymaga zainstalowania na urządzeniu i jest zależna od systemu operacyjnego i parametrów sprzętowych urządzenia. Z założenia do działania nie wymaga połączenia z Internetem, jednak w zależności od konkretnego zastosowania brak łączności może znacznie ograniczyć jej funkcjonalność. W przypadku konieczności aktualizacji takiej aplikacji, użytkownik jest zmuszony do ręcznego instalowania nowych komponentów na każdym urządzeniu, na którym chce korzystać z systemu.

3.1.2. Aplikacja webowa

Jest to rodzaj aplikacji, nie wymagającej bezpośredniego instalowania na urządzeniu ponieważ komunikacja z użytkownikiem odbywa się poprzez okno przeglądarki internetowej. Aplikacja webowa umożliwia pracę z różnych urządzeń, niezależnie od lokalizacji, a jedynym jej ograniczeniem jest dostęp do Internetu. Aktualizacja aplikacji odbywa się w większości przypadków bez udziału użytkownika. Stworzenie aplikacji webowej, która sprawnie działa na małych wyświetlaczach urządzeń mobilnych i jest niezależna od rodzaju przeglądarki jest często bardzo kłopotliwe.

3.1.3. Aplikacja mobilna

Jest to rodzaj oprogramowania dedykowanego dla urządzeń przenośnych takich jak smartfon lub tablet. Aplikacje te są zależne od systemów operacyjnych (np. Android, iOS, Windows Phone) i pozwalają użytkownikowi aplikacji na pełną mobilność. Podobnie do aplikacji desktopowych do działania nie potrzebują połączenia z Internetem jednak jest ono często wymagane, aby zagwarantować pełną funkcjonalność aplikacji.

Biorąc pod uwagę, iż do poprawnego działania projektowanego systemu i tak wymagane jest stałe połączenie z Internetem oraz ze względu na pożądaną mobilność użytkownika aplikacji przy pracy z dronami, jako docelowy rodzaj aplikacji klienckiej dla projektowanego systemu wybrano aplikację mobilną.

3.2. Wybór platformy

3.2.1. Android

System operacyjny, oparty na jądrze Linuks OS, tworzony przez organizację Open Handset Alliance składającą się z 84 firm, na czele których stoi Google. Android jest najpopularniejszym mobilnym systemem, kontrolującym ponad 80% urządzeń na rynku. Zapewnia największą otwartość dla developerów oraz wiele narzędzi wspomagających rozwój aplikacji, co w efekcie wpływa na szeroki wybór funkcji dostępnych dla użytkowników. Za główną wadę systemu można uznać najgorszą stabilność w porównaniu do konkurentów.

3.2.2. iOS

System operacyjny Apple Inc. dla urządzeń mobilnych iPhone, iPod Touch oraz iPad, bazujący na systemie operacyjnym Max OS X 10,5 i tym samym jądrze Darwin. iOS kontroluje blisko 16% urządzeń mobilnych. Z punktu widzenia developerskiego ważną kwestią jest łatwa dostępność narzędzi SDK oraz konieczność uiszczenia rocznej opłaty członkowskiej w wysokości 99$ USD w przypadku chęci publikowania aplikacji. Ponadto w przypadku aplikacji płatnych Apple pobiera 30% wygenerowanego przez nie dochodu. Aplikacje przed trafieniem do App Store podlegają znacznie bardziej rygorystycznej kontroli niż ma to miejsce w przypadku systemu Google'a. iOS cechuje się dobrą stabilnością i wygodą pracy, jednak ze względu na wysokie ceny urządzeń cieszy się znacznie mniejszą popularnością od systemu Android.

3.2.3. Windows Phone

System operacyjny dla platform mobilnych opracowany przez firmę Microsoft. To co go wyróżnia to płynność i niezawodność porównywalna do systemu iOS przy jednoczesnym stosunkowo niskim koszcie urządzeń z nim współpracujących. System ten stanowi jedynie około 3% rynku urządzeń mobilnych.

Jako platformę, na którą zostanie stworzona aplikacja wizualizująca ze względu na największą popularność na rynku urządzeń mobilnych oraz na najszerszą ofertę narzędzi programistycznych i najprężniej działającą społeczność developerską wybrano system Android.

3.3. Wybór narzędzia

3.3.1. SDK

Android SDK (Source Development Kit) jest to zestaw narzędzi dla programistów przeznaczony do tworzenia aplikacji na platformę Android w języku Java. W jego skład wchodzą wymagane biblioteki, debugger, emulator, dokumentacja, przykładowe programy oraz samouczki. Składa się z dwóch części SDK Tools, wymaganej do tworzenia aplikacji niezależnie od wersji Androida oraz Platform Tools, czyli narzędzi zmodyfikowanych pod kątem konkretnych wersji systemu. SDK jest modularne, dzięki czemu cechuje się bardzo łatwą instalacją i deinstalacją potrzebnych komponentów. Jest najbardziej popularnym narzędziem programistycznym, wykorzystywanym do pisania aplikacji na platformę Android.

3.3.2. NDK

Android NDK (Native Development Kit) jest zestawem narzędzi, pozwalającym na tworzenie aplikacji na platformę Android z wykorzystaniem języka C lub C++. Aplikacje takie można uruchomić jako proces systemu Linux, bez konieczności uruchamiania ich wewnątrz maszyny wirtualnej. Wykorzystanie NDK daje programiście większy dostęp do składników systemu, jednak wymaga od niego większej wiedzy i stwarza niebezpieczeństwo uszkodzenia systemu lub urządzenia w przypadku złego wykorzystania. Najczęściej wykorzystywane jest przy programach wymagających wysokiej wydajności takich jak aplikacje o wysokiej mocy obliczeniowej jak gry lub symulatory.

3.3.3. Xamarin

Xamarin to narzędzie programistyczne, wspierane przez Microsoft, umożliwiające pisanie wieloplatformowych, natywnych aplikacji mobilnych na platformy Android, iOS oraz Windows Phone z wykorzystaniem języka C# i platformy .NET. Pozwala na dzielenie znacznej części kodu między aplikacjami napisanymi pod różne platformy, co znacznie zmniejsza koszt utworzenia i utrzymania wieloplatformowych systemów mobilnych. Korzystanie z pełnej funkcjonalności Xamarina jest płatne, a koszt licencji to 999$ rocznie. Wersja bezpłatna jest bardzo ograniczona i pozwala jedynie na tworzenie małych aplikacji, zawierających nie więcej niż 128Kb skompilowanego kodu, bez możliwości korzystania z natywnych bibliotek języków takich jak C, C++, czy Java.

3.3.4. HTML5

Istnieje również możliwość tworzenia aplikacji mobilnych z wykorzystaniem technologii webowych takich jak HTML5, CSS oraz Javascript. Aplikacja stworzona w ten sposób jest osadzana w natywnym kontenerze, pozwalającym na dystrybucję w AppStore, czy GooglePlay. Rozwiązanie takie cechuje się stosunkowo prostą implementacją, umożliwia tworzenia aplikacji wieloplatformowych, a raz napisany kod może zostać również wykorzystany do stworzenia zwykłej aplikacji webowej. Aplikacje napisane przy użyciu HTML5 nie posiadają bezpośredniej integracji ze sprzętem i takimi składnikami jak system plików, aparat, akcelerometr, czy system nawigacji satelitarnej. Cechuje je brak wielowątkowości, a ich wydajność jest zwykle gorsza od wydajności aplikacji natywnych. Dodatkowo tworzenie aplikacji w technologiach webowych wymaga często dodatkowego nakładu pracy na implementacje, niektórych elementów powszechnie dostępnych w technologiach natywnych.

Biorąc pod uwagę powyższe argumenty, uwzględniając znajomość języka Java przez autora pracy oraz powszechną dostępność materiałów szkoleniowych i wielu użytecznych bibliotek jako narzędzie wykorzystane do stworzenia aplikacji klienckiej wybrano Android SDK.

3.4. Wybór sposobu komunikacji między serwerem, a aplikacją kliencką

W celu przesłania danych przez Sieć, klient inicjuje połączenie, najczęściej z wykorzystaniem protokołu TCP (Transmission Control Protocol), z serwerem. Protokół TCP sprzęga ich adresy IP i porty punktów końcowych oraz zapewnia bezbłędne przesyłanie danych między nimi. Jednak przesłane informacje wymagają nie tylko przetransportowania, za co odpowiada warstwa transportowa TCP, ale również zrozumienia. Dlatego ponad warstwą TCP działa dodatkowo protokół aplikacji.

3.4.1. HTTP

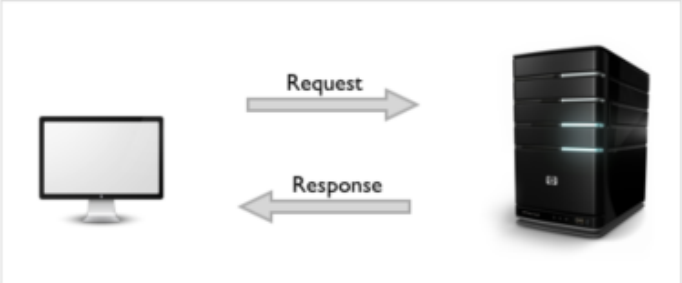
` Głównym protokołem warstwy aplikacji używanym współcześnie w przeglądarkach internetowych jest protokół HTTP. HTTP to skrót od Hypertext Transfer Protocol. Jest to protokół bezstanowy tzn. ani serwer, ani klient nie przechowują informacji o wcześniejszych zapytaniach między sobą i nie posiadają stanu wewnętrznego. Klient otwiera połączenie i wysyła komunikat żądania do serwera HTTP. Serwer następnie zwraca komunikat odpowiedzi, zawierający zasób, o który został poproszony lub kod błędu. Po dostarczeniu odpowiedzi serwer zamyka połączenie i nie przechowuje o nim żadnych informacji, co znacznie zmniejsza obciążenie serwera, jednak jest kłopotliwe w sytuacjach, w których wymagane jest zapamiętanie jakiegoś stanu związanego z konkretnym klientem. Protokół ten nie pozwala na inicjowanie połączenia z klientem ze strony serwera i co za tym idzie uniemożliwia wysyłanie wiadomości typu PUSH od serwera do klienta. Adresowanie zapytań odbywa się poprzez ujednolicony format adresowania zasobów URL (ang. Uniform Resource Location), a operacja jaką klient chce wykonać jest określana poprzez ustandaryzowane słowa kluczowe, z których najbardziej podstawowymi są :

- GET - pobieranie zasobów,
- POST - tworzenie nowych zasobów,
- PUT - aktualizowanie istniejących zasobów,
- DELETE - kasowanie istniejących zasobów

W odpowiedzi na zapytanie klienta serwer wraz z wiadomością zwrotną przesyła kod statusu zapytania. Kod statusu zapytania mówi klientowi jak interpretować odpowiedź serwera. Kody odpowiedzi są również ustandaryzowane poprzez specyfikację protokołu HTTP:

- 1xx - Wiadomość informacyjna

- 2xx - Sukces - Zapytanie zostało prawidłowo przetworzone przez serwer,
- 3xx - Przekierowanie,
- 4xx - Błąd po stronie klienta
- 5xx - Błąd po stronie serwera



Rys. 5 Schemat działania komunikacji klient-serwer, z wykorzystaniem protokołu HTTP

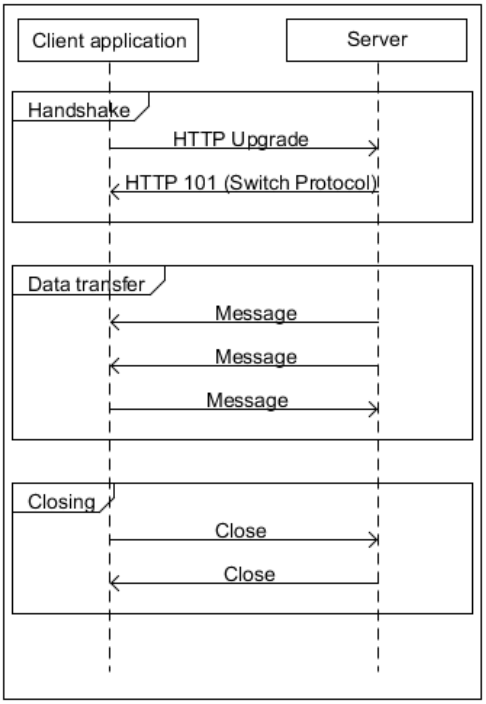
Brak możliwości wysyłania wiadomości typu PUSH ze strony serwera jest możliwy do ominięcia poprzez zastosowanie techniki nazywanej "pollingiem", która polega na stałym odpytywaniu serwera przez klienta o nowe dane. Jeśli oczekiwany zasób uległ zmianie serwer wysyła go klientowi, w innym wypadku wysyła pustą odpowiedź. Wariancją tego rozwiązania jest tzw. "Long polling", który różni się od tradycyjnego pollingu tym, że po otrzymaniu zapytania serwer w przypadku braku nowych danych nie odsyła klientowi pustej wiadomości, a utrzymuje otwarte połączenie, aż do momentu dostępności żądanego zasobu. Techniki te są jednak wysoce nieefektywne, w związku z tym, że obciążają zarówno aplikację kliencką jak i część serwerową systemu niezależnie od tego, czy dany zasób jest dostępny, czy nie.

3.4.2. WebSocket

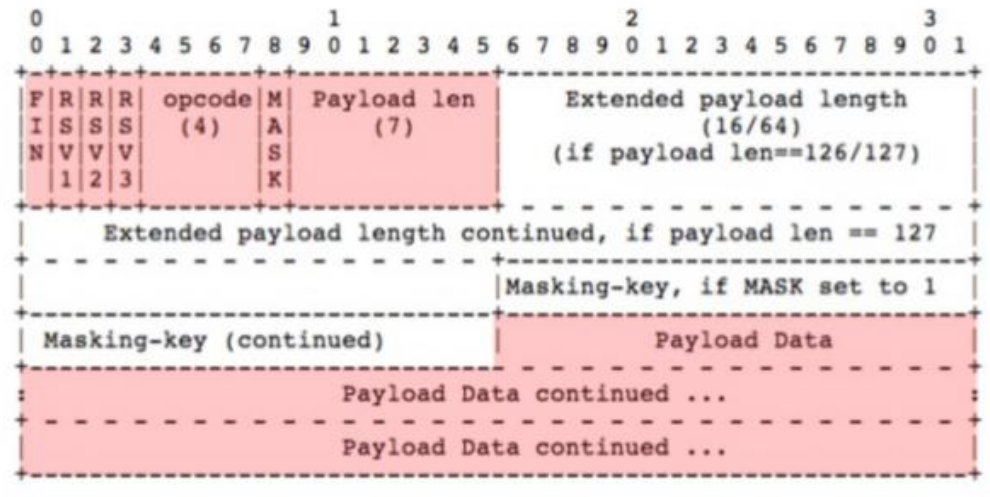
Technologia WebSocket umożliwia utworzenie trwałego, dwustronnego połączenie między klientem, a serwerem. WebSocket wykorzystuje do budowy połączenia zmianę protokołu. Klient wysyła do serwera normalne zapytanie HTTP, o specyficznym nagłówku, zawierającym klucz zabezpieczający kodowany, za pomocą algorytmu base64, który informuje serwer o chęci nawiązania połączenia poprzez protokół WebSocket. Jeśli serwer obsługuje protokól WebSocket wysyła do klienta odpowiedź, uzupełnioną o dodatkowy ciąg znaków. W ten sposób oba punkty końcowe zapewniają się, że używają protokołu WebSocket. Taki sposób nawiązywania połączenia jest powszechnie określany "uściskiem dłoni" z ang. "handshake". Jako dodatkowy mechanizm zabezpie-

czający protokół wykorzystuje szyfrowanie pakietów danych, co uniemożliwia złośliwym skryptom sterowanie serwerami proxy i wykorzystanie ich do atakowania innych użytkowników.

Po nawiązaniu połączenia za pomocą protokołu WebSocket, w przeciwieństwie do HTTP, zarówno klient jak i serwer mogą wysyłać między sobą dane, równocześnie, przez jedno połączenie TCP. Skutkuje to zmniejszeniem opóźnienia i obciążenia Sieci. Połączenia WebSocket, podobnie jak HTTP, wykorzystują standardowe porty: 80, gdy nie są szyfrowane i 443, w przeciwnym wypadku. Websocket posiada analogiczny format adresowania do HTTP: ws (Web Servieces) dla połączeń nieszyfrowanych oraz wss (Web Secure Services) dla połączeń szyfrowanych.



Rys. 6 Schemat działania połączenia, między klientem, a serwerem przy wykorzystaniu protokołu WebSocket



Rys. 7 Struktura ramki danych dla protokołu WebSocket

Jedna wiadomość może składać się z wielu ramek, z których każda ma strukturę taką, jak przedstawiono na Rys.7.

- FIN - bit identyfikujący ostatnią ramkę wiadomości,
- SRV[1-3] - 0 lub informacja o rozszerzeniu wiadomości,
- OPCODE - identyfikator typu ramki: tekstowa (1), binarna (2), kontrolna np. zamykająca połączenie (8), ping (9), pong(10),
- MASK - bit określający, czy ramka jest maskowa (dla wiadomości ze strony klienta),
- Payload len - wielkość danych:
  - Jeśli 0-125 - taka jest długość ramki,
  - Jeśli 126 - kolejne 2 bajty reprezentują 16-bitową liczbę typu unsigned integer, określającą długość ramki,
  - Jeśli 127 - kolejne 8 bajtów reprezentują 64-bitową liczbę typu unsigned integer, określającą długość ramki,
- Masking-key - zawiera 32-bitową wartość, używaną do maskowania danych (jeśli MASK 1)
- Payload Data - Zawiera przesyłane dane i dane dotyczące niestandardowych rozszerzeń, takich jak: sprecyzowany format wysyłania danych, czy wymogi semantyczne narzucone przez konkretną implementację protokołujeśli zostały one ustalone podczas nawiązywania połączenia

W tworzonym systemie założono wykorzystanie obu standardów. Komunikacja z wykorzystaniem protokołu HTTP, ze względu na łatwość implementacji zostanie wykorzystana do pobierania danych trwałych z serwera. Protokół WebSocket, w związku z koniecznością wysyłania danych

ze strony serwera do klienta, zawsze gdy ulegnie zmianie położenie drona , co jest zdarzeniem niezależnym od aplikacji klienckiej, zostanie wykorzystany jako bazowa technologia do przesyłania danych związanych z aktualizacją położenia oraz obszaru przeszukanego

3.5. Wybór formatu przesyłanych danych

3.5.1. XML

XML (ang. Extensible Markup Language), czyli uniwersalny język znaczników, jest otwartym standardem opracowanym przez World Wide Web Consortium (W3C) umożliwiającym reprezentację danych w ustrukturyzowany sposób. XML jest językiem opisującym dane, czyli meta językiem. Został stworzony z myślą o przechowywaniu danych, jednak służy również do ich transportowania oraz do tworzenia innych języków (aplikacji XML) służących do przechowywania informacji. Jest czytelny dla człowieka, co znacznie ułatwia jego wykorzystanie.

```
<?xml version="1.0" encoding="utf-8"?>
<dane>
    <user>
        <imie>jan</imie>
        <nazwisko>Kowalski</nazwisko>
    </user>
    <user>
        <imie>Piotr</imie>
        <nazwisko>Nowak</nazwisko>
    </user>
</dane>
```

Rys. 8 Przykład danych w XML

3.5.2. JSON

JSON (ang. Javascript Object Notation) jest tekstowym formatem wymiany danych opartym o literał obiektowy. Został stworzony jako lżejsza alternatywa dla XML. Wszystko co można opisać za pomocą XML'a jest również możliwe do opisu z użyciem JSON'a. Podczas gdy dane zapisane w formacie XML muszą zostać dodatkowo zinterpretowane i przekształcone na obiekty, z użyciem JSON'a dane są od razu zakodowane jako obiekt JavaScriptowy, łatwo rozumiany również przez

inne języki programowania. JSON jest mniej czytelny dla człowieka, jednak ze względu na brak znaczników jest bardziej wydajnym standardem jeśli chodzi o transportowanie oraz serializację i deserializację danych.

```
{
    "dane" : {
        "user" : [
        {
            "imie" : "Jan",
            "nazwisko" : "Kowalski"
        },
        {
            "imie" : "Piotr",
            "nazwisko" : "Nowak"
        }]
    }
}
```

Rys. 9 Przykład danych w JSON

Ze względu na wyższą wydajność oraz łatwość serializacji i deserializacji jako format przesyłanych danych wybrano JSON.

3.6. Opis API OpenStreetMap

OpenStreetMap (OSM) to globalny projekt, mający na celu stworzenie darmowej oraz swobodnie dostępnej mapy świata. Wszelkie dane zawarte w projekcie udostępniane są na otwartej licencji Open Database License (ODbL). Umożliwia to ich wykorzystanie praktycznie przez każdego, przez co OpenStreetMap jest określane mapowym odpowiednikiem Wikipedii i jest darmową alternatywą dla map komercyjnych.

Oficjalne API OpenStreetMap różni się przykładowo od API GoogleMaps tym, że udostępnia jedynie usługi do odczytu i zapisu danych w surowej formie wektorowej, podczas gdy GoogleMaps udostępnia użytkownikom konkretne usługi do rysowania i edycji map. Jednak w związku otwartością projektu OSM istnieje wiele zewnętrznych bibliotek, które sumarycznie udostępniają praktycznie każdą funkcję, którą można znaleźć w mapach komercyjnych. Ponadto dostępność do surowych danych w postaci pojedynczego pliku Planet.com, w formacie binarnym PBF lub skompresowanym XML, zawierającego wszystkie punkty, drogi i relacje tworzące mapę pozwalają na

dowolną implementację własnych usług. Dane te są aktualizowane co tydzień i możliwe do pobrania również w mniejszych paczkach dla konkretnych kontynentów, czy państw.

Mapy OpenStreetMap, podobnie jak mapy GoogleMaps generowane są w procesie nazywanym renderingiem, który konwertuje dane wektorowe na dane typu rastrowego. Na dane te zostają następnie nałożone dodatkowe style wizualne, a mapy zostają podzielone na "kafelki" i w tej formie udostępnione aplikacjom webowym i mobilnym przez zewnętrznych usługodawców. Użytkownicy OSM mają również możliwość zaimplementowania własnego sposobu renderowania map, co znacznie rozszerza możliwości programistyczne w porównaniu do GoogleMaps.

W przypadku tworzenia oprogramowania na platformę Android istnieje możliwość wykorzystania zewnętrznej biblioteki Osmdroid zapewniającej wiele przydatnych funkcji takich jak rysowanie map, ich edycja, rysowanie śladów, znaczników i wiele innych.

4. Analiza wymagań

4.1. Opis systemu

4.2. Identyfikacja aktorów

4.3. Przypadki użycia

5. Projekt systemu

5.1. Diagram komponentów

5.2. Diagram klas części serwerowej