POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: INFORMATYKA
SPECJALNOŚĆ: SYSTEMY INFORMATYKI W MEDYCYNIE

PRACA DYPLOMOWA INŻYNIERSKA

System inspekcji obszarów z wykorzystaniem autonomicznych dronów

Autonomous drone-based scouting system

AUTOR:

Mateusz Bączek

PROWADZĄCY PRACĘ:

Dr inż. Michał Kucharzak, Katedra Systemów i Sieci Komputerowych

OCENA PRACY:

WROCŁAW, 2020

Spis treści

Spis rysunków

Spis listingów

Spis tabel

Skróty

GCS (ang. *Ground control station*)
JSON (ang. *JavaScript Object Notation*)
Protobuf (ang. *Protocol Buffers*)

Rozdział 1

Wstęp

1.1. Geneza pracy

Lotnictwo autonomiczne to prężnie rozwijający się sektor branży lotniczej. Technologie pozwalające na wykorzystanie autonomicznych dronów i samolotów w nowych projektach biznesowych są dostępne na wyciągnięcie ręki - istnieją zarówno systemy zamknięte, w pełni komercyjne, jak i projekty zupełnie otwarte, pozwalające na zapoznanie się z kodem źródłowym oprogramowania sterującego statkami powietrznymi i interakcję z aktywną społecznością pasjonatów, wspólnie rozwijającą projekt.

W świecie biznesu powstają coraz to nowe rozwiązania, wykorzystujące autonomiczne maszyny do świadczenia różnorakich usług - od razu nasuwającym się rozwiązaniem jest autonomiczne dostarczanie paczek [1], ale istnieją też znacznie bardziej ambitne projekty[2]. Warto wspomnieć, że branża jest otwarta na innowatorów - firmy takie jak Boeing i Lockheed Martin sponsorują międzynarodowe konkursy przeznaczone dla młodych konstruktorów [3].

Zainteresowani autonomicznym lotnictwem inwestorzy nie ograniczają się do prywatnych firm. Rząd australijskiego stanu Queensland współorganizuje *UAV Challenge* - zawody skupione wokół rozwijania systemów wspierających służby medyczne [4].

Wykorzystanie otwartych technologii skupionych wokół awiacji autonomicznej i połączenie ich z nowoczesnymi praktykami wdrażania oprogramowania to temat atrakcyjny zarówno z perspektywy inżynierii oprogramowania jak i z perspektywy biznesowej.

Szczególnie interesujące są zagadnienia integracji komponentów systemu, oraz testowanie - które w przypadku systemu angażującego rzeczywiste maszyny nie może ograniczyć się jedynie do standardowych testów jednostkowych.

1.2. Cel pracy

Celem pracy jest stworzenie prototypu systemu monitorującego, wykorzystującego autonomiczne drony. System ma integrować się z już istniejącym oprogramowaniem sterującym autonomicznymi maszynami oraz wykorzystywać napisaną na potrzeby pracy infrastrukturę służącą do planowania tras lotów, przechwytywania, przekazywania i wyświetlania telemetrii oraz rozpoznawania obiektów na zdjęciach wykonanych w czasie lotu za pomocą sztucznej inteligencji.

Architektura systemu musi pozwalać na zautomatyzowanie procesu wdrażania systemu, oraz zautomatyzowanie wdrażania nowych funkcjonalności - każde z wdrożeń musi być poprzedzone testami integracyjnymi na poziomie całego systemu.

Prototyp ma być w pełni testowalny, zarówno na poziomie pojedynczych elementów systemu jak i na poziomie integracji całego projektu - testy muszą angażować wszystkie komponenty systemu, uruchomione wewnątrz w pełni zautomatyzowanego środowiska testowego.

1.3. Zakres pracy

Zakres pracy obejmuje elementy projektu związane z inżynierią i architekturą oprogramowania - proces projektowania struktury systemu, wybór technologii, zaprojektowanie punktów stykowych w systemie, automatyzacja procesu wdrażania systemu i nowych funkcjonalności.

Praca opisuje też sposób testowania systemu - od weryfikacji poprawności działania poszczególnych komponentów, po pełne automatyczne testy integracyjne, wykorzystujące wszystkie komponenty systemu wraz ze zintegrowanym symulatorem drona.

Rozdział 2

Wymagania funkcjonalne systemu

2.1. Oprogramowanie na dronie

Wielowirnikowce podłączone do systemu muszą być zdolne do autonomicznego lotu - w kontekście pracy oznacza to zdolność do automatycznego startu, lądowania, stabilizacji oraz samodzielnego lotu do koordynatów GPS. W trakcie lotu, maszyna musi zbierać i wysyłać dwa rodzaje danych:

- dane telemetryczne,
- zdjęcia wykonane w czasie lotu

Dane telemetryczne muszą zawierać informacje o pozycji drona oraz identyfikować maszynę za pomocą unikatowego identyfikatora oraz numeru lotu. Przesyłany jest też poziom naładowania baterii oraz informacja, czy w obecnym czasie prowadzone jest nagrywanie.

2.2. Protokoły wymiany danych

W przypadku systemu działającego autonomicznie, wymiana danych jest kluczowym elementem pozwalającym na sprawdzanie poprawności działania i diagnozowania błędów w systemie. Podczas lotów testowych często nie ma możliwości bezpośredniej obserwacji systemu lub ingerencji w jego sposób działania. Odpowiednia architektura zbierająca i archiwizująca dane z lotów pozwala znacznie szybciej wykryć potencjalne problemy i zapobiec krytycznym błędom.

2.2.1. Dane telemetryczne

Protokół do wymiany danych telemetrycznych powinien wysyłać dane w postaci binarnej, gdyż jest to bardziej efektywne niż kodowanie ich w postaci tekstowej (na przykład w formacie `JSON`, typowym dla języków wysokopoziomowych - wykorzystywanym w technologiach webowych).

Protokół powinien być w łatwy sposób rozszerzalny, pozwalając w przyszłości zredefiniować część wysyłanych pakietów lub dodać nowe dane, bez tracenia kompatybilności wstecznej bądź konieczności przebudowania całego systemu. Poszczególne komponenty systemu będą pisane

w różnych językach programowania - biorąc to pod uwagę, pożądaną cechą protokołu jest też możliwość szybkiego przeportowania go na inny język programowania.

2.2.2. Zdjęcia wykonane w trakcie lotu

Efektywny przesył zdjęć oraz filmów to temat zbyt obszerny i wymagający, żeby poruszać go w treści pracy - system powinien wykorzystywać dowolny prosty w implementacji protokół wysyłania zdjęć. Architektura systemu powinna zapewnić możliwość prostej wymiany tego komponentu, dzięki czemu w przyszłości będzie możliwe zastąpienie go przez bardziej zoptymalizowane rozwiązanie.

2.3. Oprogramowanie serwerowe

Serwer webowy jest komponentem, który odbiera, archiwizuje i przekazuje dane nadchodzące z dronów do aplikacji klienckiej. Jest punktem centralnym systemu, wykorzystywanym bezpośrednio przez wszystkie pozostałe elementy.

2.3.1. Odbiór i multipleksowanie telemetrii

Aby umożliwić diagnozowanie stanu systemu w czaie rzeczywistym - szczególnie stanu wykonujących lot wielowirnikowców, telemetria nadchodząca z maszyn nie może być jedynie archiwizowana na dysku serwera centralnego. Konieczną funkcjonalnością jest przesyłanie jej w czasie rzeczywistym do wielu jednocześnie podłączonych klientów.

Umożliwi to podjęcie akcji w przypadku wykrycia krytycznego błędu, który mógłby zakończyć się uszkodzeniem bądź rozbiciem drona, ułatwi też wykonywanie testów - zarówno na rzeczywistych maszynach, jak i wykorzystujących symulatory lotu.

2.4. Oprogramowanie klienckie

Aplikacja kliencka skupiona jest wokół trzech funkcjonalności:

1. planowanie tras i harmonogramu lotów,
2. odbiór telemetrii i zdjęć z dronów w czasie rzeczywistym,
3. przegląd i analiza telemetrii oraz zdjęć zarchiwizowanych z poprzednich lotów.

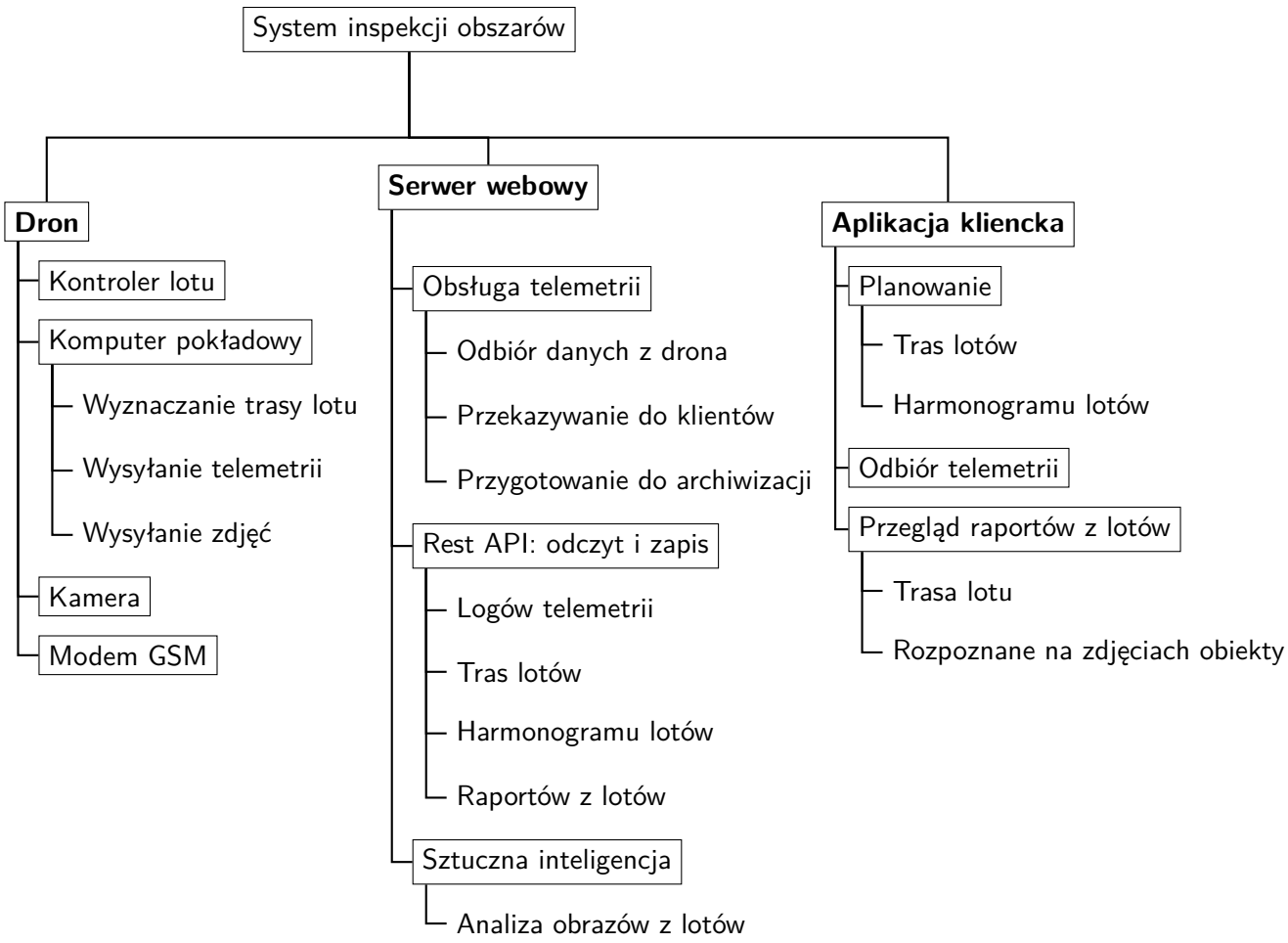
Kluczowym elementem aplikacji klienckiej jest obsługa mapy - wszystkie wymienione funkcjonalności wymagają wizualizacji nadchodzących danych geograficznych, rozszerzonych o dodatkowe informacje (na przykład godzina przelotu przez dany punkt lub zarejestrowane w danym miejscu zdjęcia i wykryte na nich obiekty).

Rozdział 3

Architektura systemu: przegląd i wybór technologii

3.1. Zarys architektury

Rys. 3.1: Zarys architektury systemu



3.2. Dron

3.2.1. Kontroler lotu

Wielowirnikowce podłączone do systemu, muszą być wyposażone w kontroler lotu – umożliwiający autonomiczny lot, stabilizację oraz obsługę peryferiów takich jak czujniki oraz silniki.

Spośród aktywnie rozwijanych i popularnych projektów [5] tworzących oprogramowanie do kontrolerów lotu, można wyróżnić cztery najpopularniejsze inicjatywy:

Tab. 3.1: Najpopularniejsze otwarte projekty oprogramowania obsługującego kontrolery lotu

Nazwa projektu	**Rok założenia**	**Docelowy hardware**	**Licencja**
ArduPilot[6]	2009	otwarte mikrokontrolery ARM	GPLv3
AutoQuad[7]	2011	mikrokontrolery STM Cortex M4	GPLv3
LibrePilot[8]	2015	zamknięte źródłowo kontrolery lotu, bazujące na architekturze ARM	GPLv3
PX4 Autopilot[9]	2012	otwarte mikrokontrolery ARM	BSD

3.2.2. Komputer pokładowy

Poza kontrolerem lotu, który zawiera jedynie oprogramowanie niezbędnie do sterowania lotem i udostępniania strumienia telemetrii (zazwyczaj przez port szeregowy), na maszynie musi znaleźć się też drugi komputer – do zastosowań bardziej ogólnych: komunikacja z bardziej wysokopoziomowymi peryferiami, takimi jak kamera lub modem GSM.

3.3. Protokoły wymiany danych

3.4. Oprogramowanie serwerowe

3.5. Oprogramowanie klienckie

3.6. Struktura repozytoriów

3.7. Wspólne punkty stykowe - `git submodules`

Rozdział 4

Wdrażanie systemu

4.1. Konteneryzacja

4.2. Automatyczne budowanie projektów

4.3. Automatyczne aktualizacje kontenerów

Rozdział 5

Testy systemu

5.1. Testy jednostkowe

5.2. Testy integracyjne

5.2.1. Symulacja i symulatory

5.3. Systemy ciągłej integracji

5.4. Testy w terenie

Rozdział 6

Podsumowanie

6.1. Wyniki testów

6.2. Osiągnięta sprawność

6.3. Pola do poprawy

6.4. Wnioski

Literatura

[1] Amazon Inc, "Amazon prime air," 2013. https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011.

[2] A. Claesson, A. Bäckman, M. Ringh, L. Svensson, P. Nordberg, T. Djärv, and J. Hollenberg, "Time to Delivery of an Automated External Defibrillator Using a Drone for Simulated Out-of-Hospital Cardiac Arrests vs Emergency Medical Services," *JAMA*, vol. 317, pp. 2332–2334, 06 2017.

[3] R. Pogrzebny and K. Florencka, "Sukces polskich studentów na zawodach sae aero design w usa," 2018. https://naukawpolsce.pap.pl/aktualnosci/news%2C29012%2Csukces-polskich-studentow-na-zawodach-sae-aero-design-w-usa.html.

[4] UAV Challenge , "Sponsors and supporters 2019 & 2020," 2019. https://uavchallenge.org/about/sponsors-and-supporters/.

[5] E. S. M. Ebeid, M. Skriver, and J. Jin, "A survey on open-source flight control platforms of unmanned aerial vehicle," 08 2017.

[6] "Ardupilot - strona domowa projektu," 2020. https://ardupilot.org/.

[7] "Autoquad - historia projektu," 2020. http://autoquad.org/home/autoquad-project-timeline/.

[8] "Librepilot - strona domowa projektu," 2020. https://www.librepilot.org/site/index.html.

[9] "Px4 autopilot - strona domowa projektu," 2020. https://px4.io/.