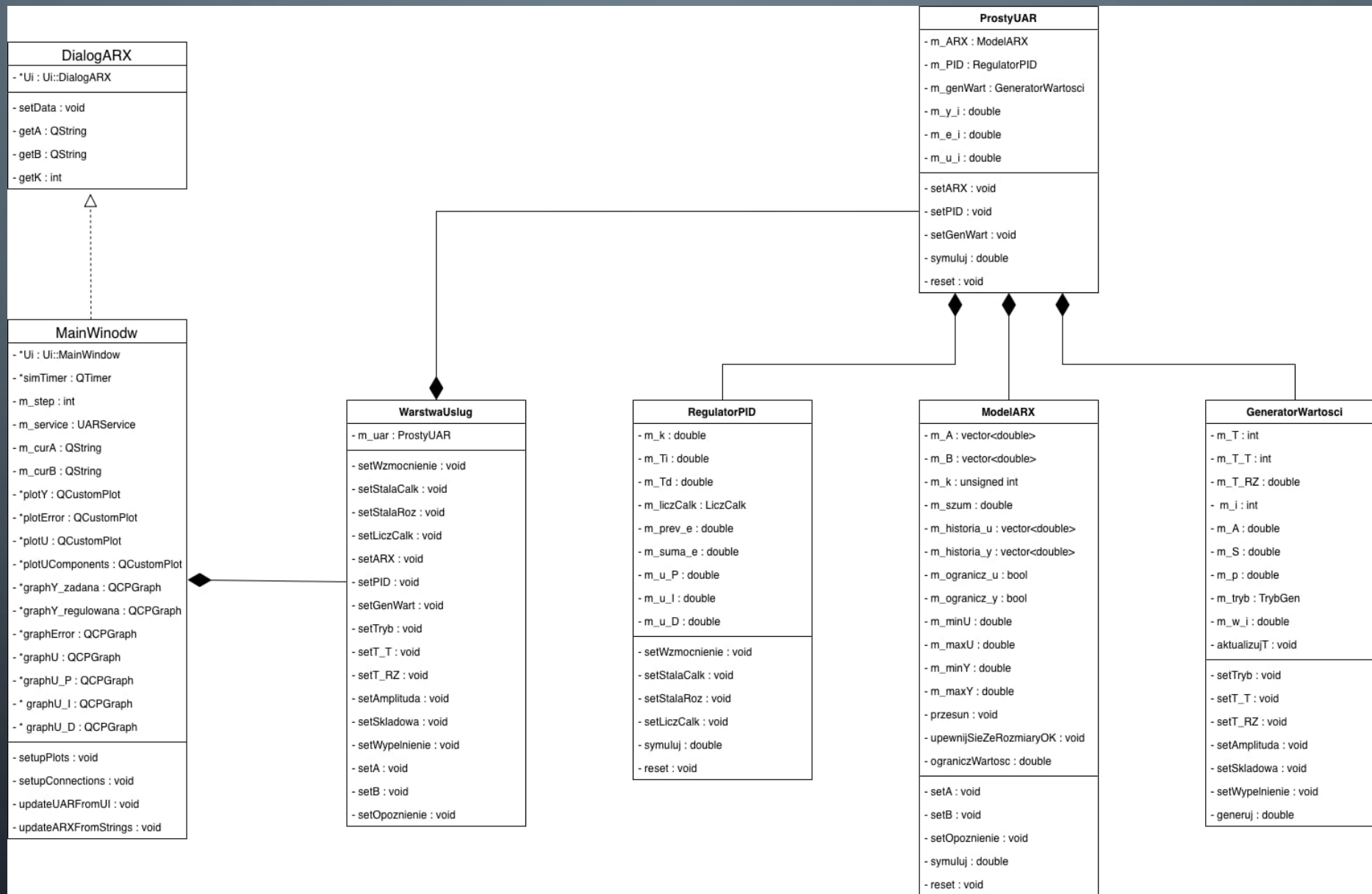




PROGRAMOWANIE KOMPUTERÓW PROJEKT (SPOTKANIE III)

SKŁAD SEKCJI:
ARTUR ZABOR,
MICHAŁ WALTEROWSKI

SCHEMAT UML



FILM OBRAZUJĄCY DZIAŁANIE APLIKACJI

NAJWIĘCEJ KŁOPOTÓW

1. Wybór i implementacja biblioteki do wykresów (QCustomPlot)

Największą trudnością było znalezienie narzędzia, które posłuży do narysowania wykresów.

Dlaczego to było trudne? Standardowe biblioteki Qt nie posiadają wbudowanego, zaawansowanego modułu do wykresów naukowych. Wybór QCustomPlot wymagał ręcznego zarządzania pamięcią (dynamiczne tworzenie obiektów `new QCustomPlot()`) oraz zrozumienia specyficznego API biblioteki (np. zarządzanie warstwami, legendami i dynamicznym skalowaniem osi `rescaleAxes()`).

2. Komunikacja między oknami ARX i MainWindow

Połączenie okna dialogowego DialogARX z głównym oknem aplikacji

Dlaczego to było trudne? Trudność polegała na bezpiecznym przekazaniu danych w obie strony. Musieliśmy zapewnić, że po kliknięciu "OK" w popupie, parametry (ciągi znaków `a` i `b`) zostaną poprawnie przesłane do usługi UARService, nie zawieszając przy tym głównego wątku symulacji.

NAJWIĘKSZA SATYSFAKCJA

1. Pierwsza działająca wersja

Nic nie daje takiej satysfakcji jak chwila, w której po kliknięciu przycisku START, na wykresie pojawia się pierwsza linia. To moment, w którym teoretyczne równania różnicowe modelu ARX i algorytm PID stają się namacalnym, działającym procesem.

2. Ułożenie interfejsu

Dopracowanie pliku .ui tak, aby wszystkie kontrolki były logicznie pogrupowane (Generator, PID, System), daje poczucie satysfakcji. Użycie QGridLayout do równego rozmieszczenia czterech wykresów sprawiło, że aplikacja zaczęła wyglądać znacznie lepiej.

REGULATOR PID -
PARAMETRY

GENERATOR WARTOSCI
ZADANEJ

PARAMETRY SYMULACJI

USTAWIENIA ARX

ZAPIS

ODCZYT

START

STOP

RESET

WYKRESY

STATUS SYMULACJI

OMÓWIENIE MODYFIKACJI WSTĘPNEGO PROJEKTU GUI

ARX

WEKTOR A :

A1:

A2:

A3:

WEKTOR B :

B1:

B2:

B3:

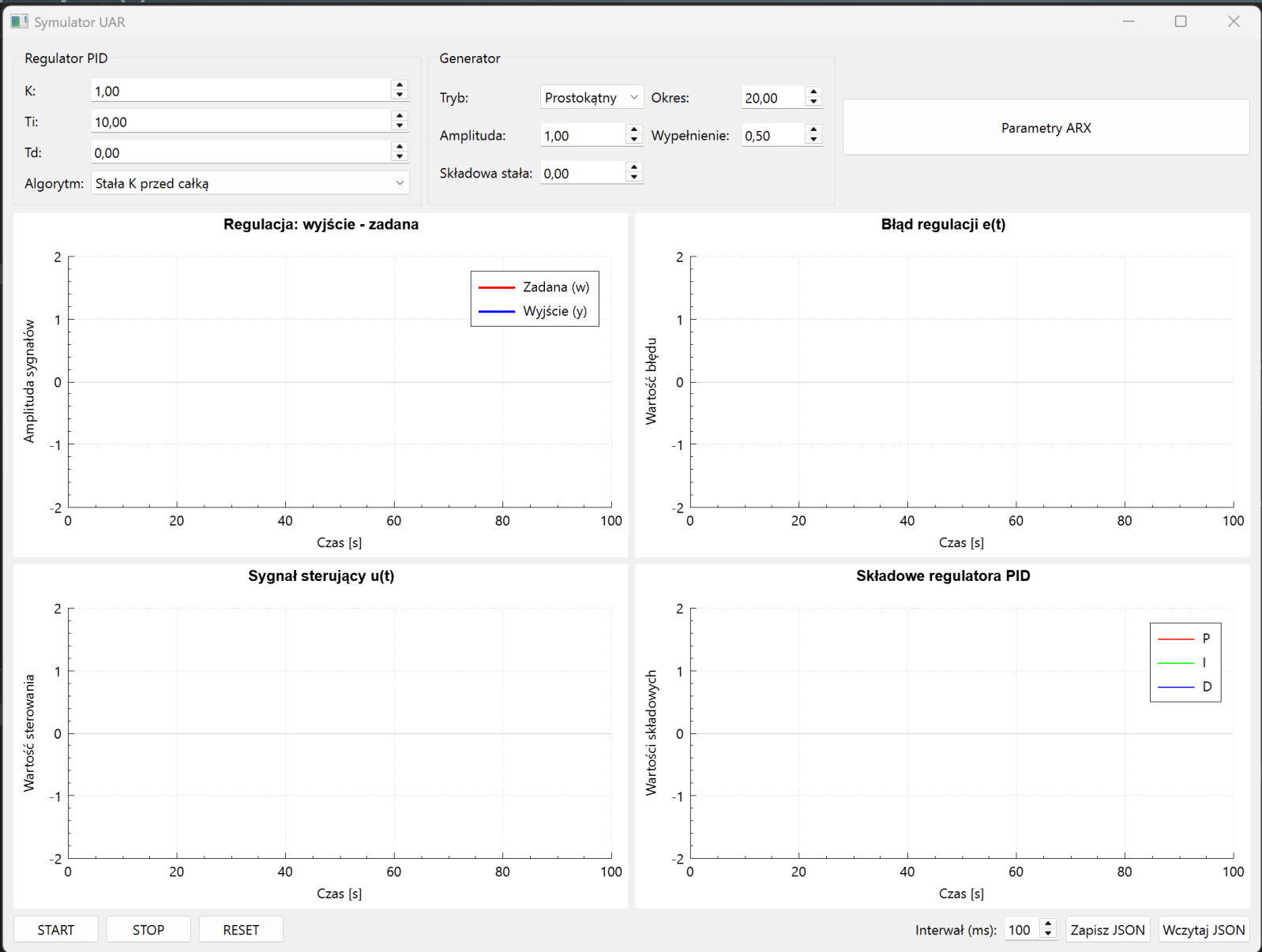
Umin:
Umax:

Ymin:
Ymax:

ZAKŁUCENIA:

OPÓŹNIENIE:

ZAPISZ



Ustawienia ARX

Wielomian A (np. 0.1, 0.2): 0.1, 0.2

Wielomian B (np. 1.0, 0.5): 1.0, 0.5

Opóźnienie k: 1

OK Cancel

KOMUNIKACJA WARSTWY PREZENTACJI Z WARSTWĄ USŁUG

```
// Warstwa usług zarządzająca logiką  
UARService m_service;
```

Warstwa prezentacji posiada obiekt usługi na wyłączność. Dzięki temu GUI nie musi znać szczegółów obliczeń, a jedynie publiczne metody klasy UARService.

```
void MainWindow::simulateStep() {  
    updateUARFromUI();  
    SimulationData d = m_service.nextStep(m_step);  
    graphY_zadana->addData(d.x, d.setpoint); graphY_regulowana->addData(d.x, d.y);  
    graphError->addData(d.x, d.error); graphU->addData(d.x, d.u);  
    graphU_P->addData(d.x, d.uP); graphU_I->addData(d.x, d.uI); graphU_D->addData(d.x, d.uD);  
  
    for (QCustomPlot* p : {plotY, plotError, plotU, plotUComponents}) {  
        p->xAxis->setRange(m_step > 100 ? m_step - 100 : 0, m_step + 1);  
        p->rescaleAxes(true); p->replot();  
    }  
    m_step++;  
}
```

Metoda simulateStep wywołuje m_service.nextStep(). MainWindow nie wykonuje żadnych obliczeń – przekazuje to zadanie do UARService.

The background is a dark blue gradient. In the corners, there are white, stylized circuit-like lines with small circles at the ends, resembling a network or data flow diagram.

DZIĘKUJEMY ZA UWAGĘ