Projektbezeichnung: Tytuł projektu:	[SmartRiver: Intelligentes Odergebiet/SmartRiver: Inteligentne Nadodrze]
Antragsnummer: Numer wniosku:	[85029892]
Output / Produkt:	Dokument: **„Datenerfassung und -verarbeitung"** / Dokument **„Gromadzenie i przetwarzanie danych"**

Dotyczy: działanie nr 6 – **Gromadzenie i przetwarzanie danych**

Opis działania: W ramach tego działania zostanie wybrane, zaimplementowane i przetestowane rozwiązanie gwarantujące zbieranie danych oraz analizowanie i prezentowanie ich w systemie.

Autorzy dokumentu: IHP, UZ

Odbiorcy dokumentu: Użytkownicy końcowi

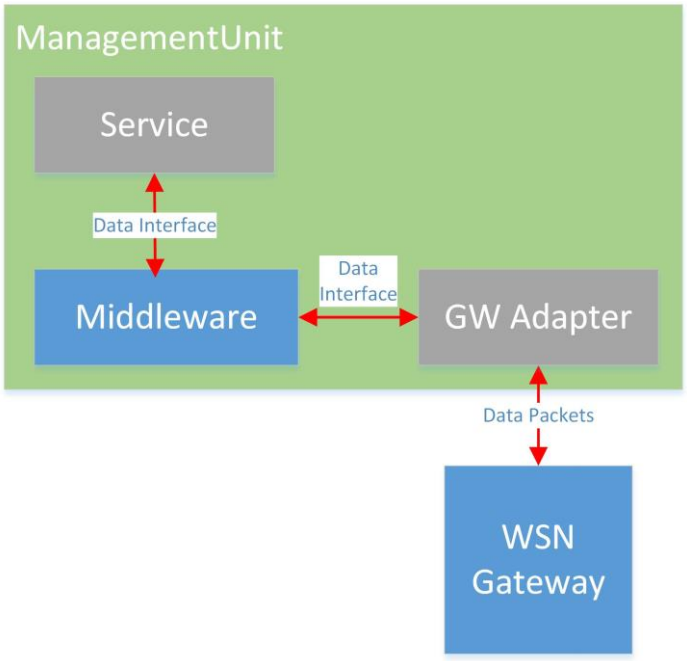
Spis treści

1. Wstęp

Jednym z wymagań projektowanego systemu jest przechowywanie i przetwarzanie danych zebranych z czujników na monitorowanym obszarze. W celu zrealizowania tego wymagania, wykorzystano platformę SmartDSM, która składa się z części przechowującej (middleware) oraz przetwarzającej (serwisy). Realizacja przechowywania i przetwarzania danych w platformie SmartDSM pozwala na pełną kontrolę nad strukturą, przetwarzaniem oraz dostępem do danych. Platforma oferuje dodatkowe narzędzia do zarządzania, zapewniania bezpieczeństwa oraz usługi odnajdowania urządzeń i danych w sieci. Platforma ta jest propozycją dla implementacji rozwiązań w obszarze Smart City.

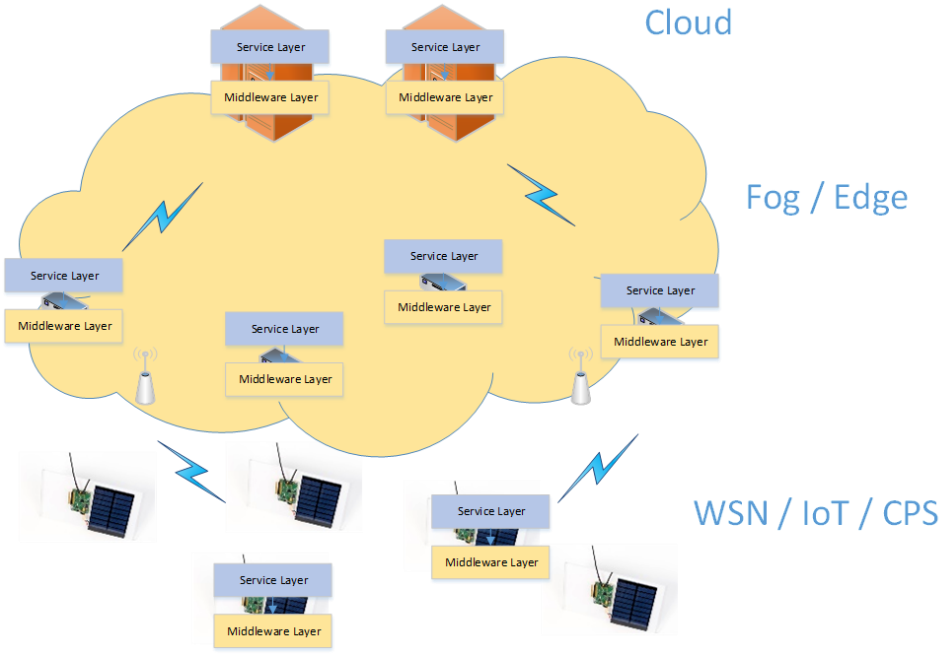
2. Platforma SmartDSM

Platforma składa się z modułów, które można wykorzystać do stworzenia rozwiązania umożliwiającego przechowywanie i przetwarzanie danych. Została stworzona w technologii Java, co pozwala na uruchamianie w kontekście wielu systemów operacyjnych. Bezpieczeństwo komunikacji zapewnia infrastruktura klucza publicznego (PKI), która została zaimplementowana przy użyciu rozszerzenia Java Secured-Socket Extension (JSSE). Każdy z uczestników komunikacji posiada certyfikat, który jest używany do szyfrowania wiadomości oraz identyfikacji uczestnika. Podstawowe moduły platformy to serwer middleware (przechowywanie danych) oraz biblioteka (oferująca interface Data Interface) umożliwiająca komunikację z serwerem.



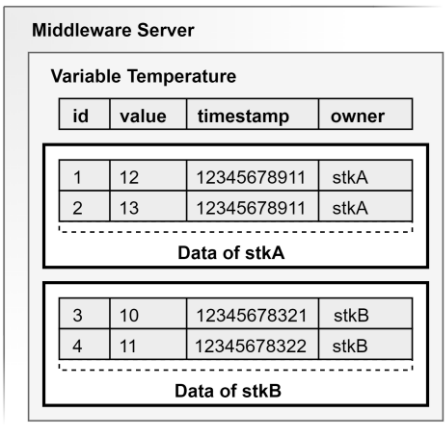
Rys. 1. Podstawowe moduły platformy SmartDSM w systemie Smart River

Funkcjonalności systemu, który bazuje na platformie SmartDSM, nazywane są serwisami. Implementuje się je jako osobne aplikacje Java korzystające z biblioteki Data Interface. Biblioteka zapewnia połączenie z wybranym serwerem middleware oraz umożliwia wywoływanie funkcjonalności związanych z przechowywaniem danych (zapis, odczyt, aktualizacje, usuwanie) oraz komunikacją z usługami platformy, takimi jak: inne serwery middleware, odnajdowanie (Discovery), pośrednia wymiana danych (Proxy) oraz weryfikacja uczestników (Certification Authority). Biblioteka umożliwia tworzenie tzw. subskrypcji, co pozwala na otrzymywanie powiadomień, przykładowo gdy wybrana wartość ulegnie zmianie lub nastąpi zapis oczekiwanej wartości. Dodatkowo biblioteka umożliwia zarządzanie pozwoleniami (w zakresie własnych danych lub wszystkich – posiadając odpowiednie pozwolenie) na wybranym serwerze middleware oraz pobieranie statusu serwera. Sposób implementacji platformy pozwala na zastosowanie wielu scenariuszy hierarchii, w zależności od wymagań, może zostać użyty jeden centralny serwer middleware lub wiele rozproszonych serwerów komunikujących się ze sobą.



Rys. 2. Przykładowa struktura sieci rozproszonej

Operacje na danych realizowane są w oparciu o zmienne i właścicieli. Zmienna w kontekście platformy to obiekt definiujący listę uporządkowanych ciągów wartości (krotka). Każda zmienna posiada swoją nazwę, limity oraz definicję pól. Do każdej zmiennej właściciel może zapisać dowolną ilość ciągów wartości (krotek), które mogą reprezentować dowolne dane (np. wilgotność, temperatura, itp.). W zależności od potrzeb, krotki mogą przechowywać słowa kontrolne używane do sterowania np. urządzeniami w systemie, wartości historyczne wyników pomiarów lub informacje o strukturze sieci. Właściciel w kontekście platformy to podmiot, który zapisuje dane w zmiennych przy użyciu serwisów (komunikujących się z serwerem middleware za pomocą biblioteki Data Interface). Każdy serwis działa w imieniu swojego właściciela, w którego imieniu wykonywane są akcje na serwerze middleware.



Rys. 3. Zmienna przechowująca wyniki pomiarów temperatury

Rysunek 3 przedstawia przykładową zmienną przechowującą wyniki pomiarów temperatury dwóch właścicieli (stkA oraz stkB). Na serwerze istnieje jedna zmienna Temperature, która przechowuje wszystkie wartości każdego z właścicieli.

2.1. Uprawnienia

Domyślnie, dostęp do wartości zapisanych w zmiennej posiada jedynie właściciel, w którego imieniu serwis zapisał wynik. Serwis w imieniu właściciela poprzez bibliotekę Data Interface może nadać lub odebrać dostęp. Gdy dostęp zostanie nadany, inni właściciele (przy użyciu serwisów) mogą np. odczytywać dane lub tworzyć subskrypcje, aby otrzymywać powiadomienia gdy ktokolwiek posiadający uprawnienia wykona operację na zmiennej (zapis, odczyt, aktualizacja, usunięcie). Tabela 1 przedstawia przykładowe pozwolenie na odczyt wartości temperatury właściciela stkA przez dowolny serwis właściciela stkB, bez ograniczeń czasowych pomiędzy kolejnymi odczytami. Rys. 4 przedstawia przykładowe użycie biblioteki Data Interface w celu utworzenia pozwolenia, które przedstawia Tabela 1.

Tabela 1. Przykładowe pozwolenie na odczyt wartości (wybrane pola)

id	permission	owner	service	stakeholder	min_delay
1	variables.variable.Temperature .read	stkA	*	stkB	0

```
DataInterface dataIntf = <połączenie z serwerem middleware, jako stkA>
DataInterfaceRequest request = dataInf.createRequest();
String permission = "variables.variable.Temperature.read";
request.addPermission(permission, "stkA", "*", "stkB", ..., 0);
DataInterfaceResponse response = request.execute();
PolicyResponse policyResponse = response.getFirstWithType(PolicyResponse.class);

// Wynik operacji dostępny w zmiennej policyResponse
```

Rys. 4. Przykładowe zezwolenie na dostęp (biblioteka Data Interface)

Każdy serwer middleware posiada domyślny zestaw pozwoleń uprawniający każdego właściciela do wykonywania podstawowych operacji na serwerze, np. tworzenie zmiennych, pobieranie listy pozwoleń lub subskrypcji. Administrator może dowolnie modyfikować pozwolenia, w tym definiować pozwolenia domyślne. Administracja odbywa się poprzez aplikację internetową AdminGUI, dostępną w każdym serwerze middleware.

2.2. Operacje na danych

W projekcie Smart River, warstwa middleware jest odpowiedzialna za przechowywanie i przetwarzanie danych. Danymi mogą być wyniki pomiarów lub sygnały sterujące. W zależności od typu danych, przewidziano odpowiednie scenariusze, zgodnie z którymi zaimplementowano przesyłanie, przetwarzanie i przechowywanie danych.

Modyfikowanie danych odbywa się poprzez wywoływanie odpowiednich metod biblioteki Data Interface, która zapewnia połączenie z wybranym serwerem middleware.

```
DataInterface dataIntf = <połączenie z serwerem middleware, jako stkA>
DataInterfaceRequest request = dataInf.createRequest();
String permission = "variables.variable.Temperature.read";
request.addPermission(permission, "stkA", "*", "stkB", ..., 0);
String[] keys = new String[] { "accuracy" };
Object[] values = new Object[] { 0.5 };
request.write("Temperature", "stkA”, <serwer>, 22, keys, values).as("write");
request.read("Temperature", "stkB", <serwer>, <filtr>).as("read");
DataInterfaceResponse res = request.execute();
VariableResponse wr = res.getFirstNamedAs("write", VariableResponse.class);
Class<VariablePageResponse> rr_class = VariablePageResponse.class;
VariablePageResponse rr = res.getFirstNamedAs("read", rr_class);

// Wyniki operacji dostępne w zmiennych wr oraz rr
```

Rys. 5. Przykładowy zapis oraz odczyt danych

Metody związane z odczytem danych oraz pobieraniem listy obiektów (zmienne, pozwolenia, subskrypcje) posiadają możliwość stronicowania, filtrowania oraz sortowania, co pozwala na pobranie dowolnych wartości w wybranej kolejności.

W celu otrzymywania powiadomień o modyfikacji wartości zmiennych przez innych właścicieli, należy zaimplementować interfejs SubscriptionListener, przekazać referencję do instancji Data Interface oraz utworzyć subskrypcję na serwerze middleware. Rysunek 6 obrazuje działanie subskrypcji na platformie SmartDSM. Początkowo utworzony zostaje obiekt Data Interface, następnie przekazywany jest obiekt typu SubscriptionListener. Końcowo tworzona jest subskrypcja. Po utworzeniu subskrypcji, jeśli właściciel posiada dostęp do danych, na które została utworzona subskrypcja, metoda onSubscription zostanie wywołana każdorazowo gdy nastąpi zapis wartości do zmiennej. Gdy został przekazany filtr, metoda zostanie wywołana tylko w przypadku, gdy wartość spełnia wymagania przekazanego filtru.

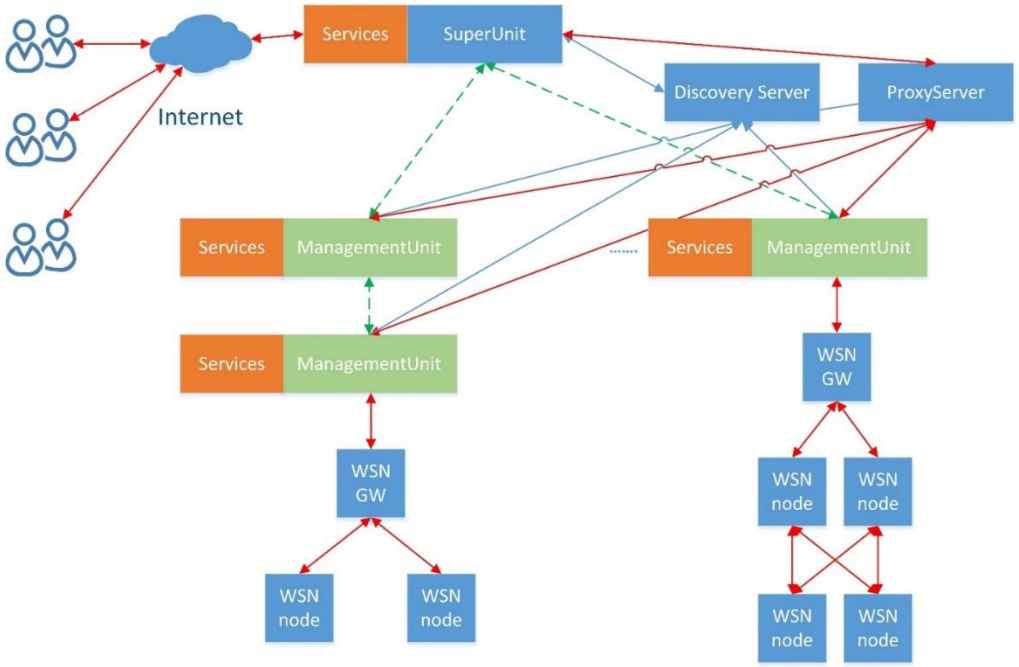
```
DataInterface dataIntf = DataInterface.create(...)
.registerSubscriptionListener(new SubscriptionListener() {
  public void onSubscription(SubscriptionInfo info) {
    System.out.println("Otrzymano powiadomienie: " + info);
  }
})
...
DataInterfaceRequest request = dataInf.createRequest();
String permission = "variables.variable.Temperature.read";
request.subscribe(VariableEvent.WRITE, <nazwa zmiennej>, <nazwa właściciela>,
<serwer docelowy>, <filtr>);
DataInterfaceResponse response = request.execute();
SubscriptionResponse sr = response.getFirstWithType(SubscriptionResponse.class);

// Wynik tworzenia subskrypcji dostępny w zmiennej sr
```

Rys. 6. Tworzenie subskrypcji oraz otrzymywanie powiadomień

3. Scenariusze

Rys. 7 przedstawia architekturę systemu w projekcie Smart River, do którego odnoszą się Sekcje 3.1 – 3.4. Źródłem danych są węzły WSN Node, posiadające sensory dla każdej monitorowanej wartości w systemie. Wyniki pomiarów przechowywane są na serwerach middleware. Nadrzędny serwer middleware (SuperUnit) udostępnia aplikację internetową użytkownikom końcowym poprzez Internet. W sieci dostępne są usługi, takie jak odnajdowanie (Discovery) i komunikacja pośrednia (Proxy).

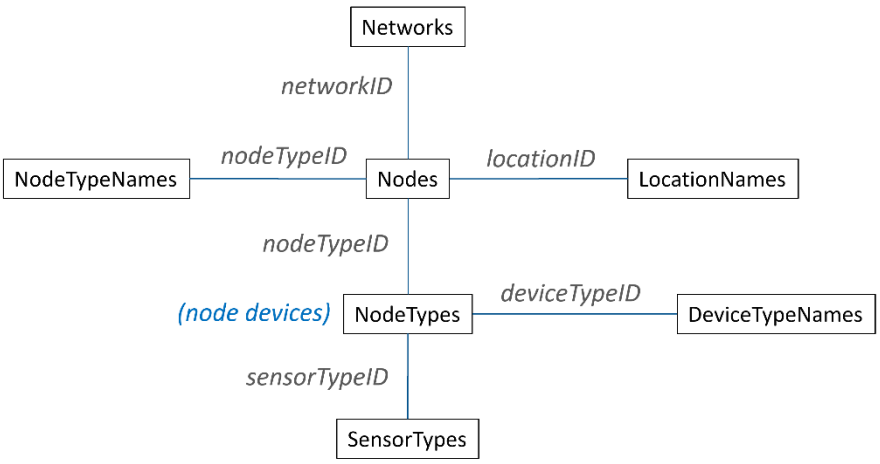


Rys. 7. Architektura systemu pomiarowego

3.1. Przechowywanie informacji o sieci

Rys. 8 przedstawia strukturę sieci Smart River przechowywaną w zmiennych na serwerze middleware. Zmienna Networks przechowuje informacje o sieciach wchodzących w skład systemu (Frankfurt i Słubice). Każda sieć posiada listę węzłów (Nodes). Każdy węzeł posiada przypisaną informację o lokalizacji (LocationNames), typie węzła (NodeTypeNames) oraz listę urządzeń (NodeTypes). Każde urządzenie posiada informację o typie urządzenia (DeviceTypeNames) oraz listę sensorów (SensorTypes).

W ramach projektu utworzono narzędzie (Sekcja 4), które pozwala na pobieranie, aktualizowanie, eksportowanie oraz wizualizację konfiguracji.



Rys. 8. Relacje zmiennych przechowujących informacje o strukturze systemu

3.2. Przesyłanie wyników pomiarów

W zależności od mierzonych parametrów, węzły WSN Node wysyłają wyniki pomiarów do węzła-bramy (WSN GW) zgodnie z częstotliwością dla aktualnego trybu pracy systemu (normalny, badawczy, alarmowy, testowy i serwisowy). Następnie, węzły WSN GW wysyłają wyniki pomiarów do serwisów odpowiedzialnych za obsługę węzła GW. Serwisy obsługujące zapisują wyniki na serwerach middleware (Sekcja 2.2). Każda wartość mierzona jest przechowywana w osobnej zmiennej, która posiada dodatkowe pola opisujące dany wynik pomiaru – położenie geograficzne, identyfikator węzła, urządzenia oraz sensora, który wykonał pomiar.

Tabela 2. Przykładowy wynik pojedynczego pomiaru

id	timestamp	owner	source	value	nodeId	deviceTypeId	sensorId	lat	lon
1	1653909134713	Miasto1	<źródło>	20	1	1	1	54.312	14.333

3.3. Przesyłanie konfiguracji i sygnałów sterujących

Konfiguracja i sygnały sterujące przechowywane są w zmiennych na serwerze middleware. Przesyłanie odbywa się poprzez zapis następnej wartości do zmiennej. Serwisy bazujące na konfiguracji lub wykonujące operacje na podstawie sygnałów tworzą subskrypcje, co pozwala na otrzymywanie powiadomień o następnej wartości w momencie gdy jest dostępna.

Tabela 3. Przykładowa zmienna przechowująca sygnały sterujące

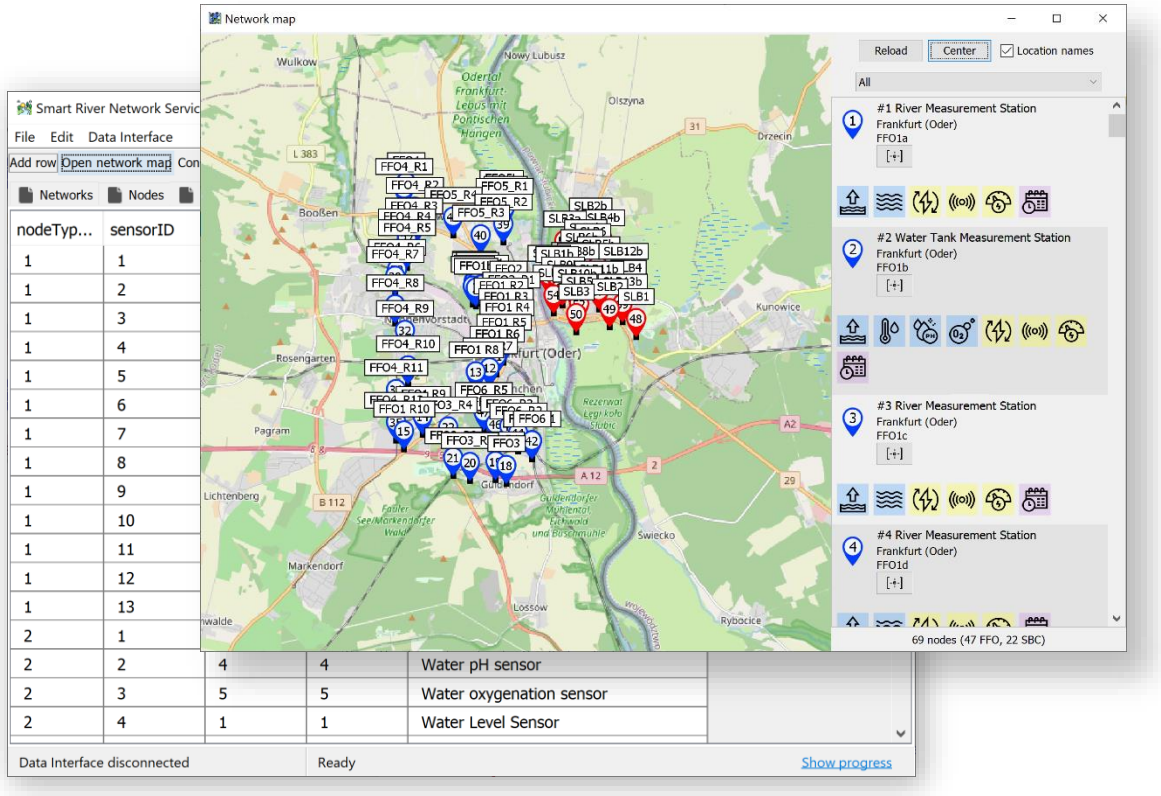
id	timestamp	owner	source	value	name	nodeId	deviceTypeId	sensorId
1	1653909134713	Miasto1	<źródło>	30000	interval	1	1	1
2	1653909149451	Miasto1	<źródło>	true	ABC_enabled	1	1	1

3.4. Przetwarzanie wyników pomiarów

Serwisy przetwarzające tworzą subskrypcje i natychmiastowo otrzymują powiadomienia gdy zostaną zapisane nowe wyniki pomiarów. Wyniki przetwarzania zapisywane są w zmiennych, dla których mogą być tworzone kolejne subskrypcje w celu otrzymywania powiadomień o przetworzonych wynikach pomiarów.

4. Narzędzia

W celu łatwiejszego zarządzania konfiguracją sieci, opracowano narzędzie pozwalające na pobieranie aktualnej konfiguracji z serwera middleware, wprowadzanie zmian, eksport oraz wizualizację. Narzędzie wykorzystuje bibliotekę Data Interface do komunikacji z serwerem przechowującym konfigurację.



Rys. 9. Okna programu do zarządzania konfiguracją sieci

5. Podsumowanie

W dokumencie przedstawiono sposób w jaki dane są gromadzone i przetwarzane w projektowanym systemie. Zaprezentowano platformę SmartDSM, sposób w jaki platforma została wykorzystana w projekcie. Przedstawiono strukturę sieci oraz metodologię używaną podczas operacji na różnych typach danych (wyniki pomiarów, konfiguracja, sygnały sterujące). Przedstawiono sposób przetwarzania wyników pomiarów oraz ich udostępniania. Dodatkowo zaprezentowano narzędzie do konfiguracji i wizualizacji sieci.

Platforma oraz sposób jej użycia pozwala na łatwe rozbudowanie systemu o nowe funkcjonalności w przyszłości.

6. Wersje dokumentu

Zmieniający	Zmiana	Wydanie	Wersja
IHP	Wersja wyjściowa z poprawkami	A	- (0)
		A	1