Politechnika Gdańska
Wydział Elektroniki, Telekomunikacji
i Informatyki

mgr inż. Jakub Wszołek

Sieciowy Monitor Obiektu – rozproszony system inteligentnego zarządzania złożonymi obiektami przemysłowymi

Rozprawa doktorska

Promotor:
prof. dr hab. inż. Zdzisław Kowalczuk
Wydział Elektroniki Telekomunikacji
i Informatyki
Politechnika Gdańska

Gdańsk, 2019

Pragnę wyrazić wdzięczność Panu Profesorowi Zdzisławowi Kowalczukowi
za pomoc w kształtowaniu koncepcji pracy, opiekę
i wsparcie doświadczeniem oraz wiedzą.
Szczególne słowa podziękowania składam mojej żonie Annie.

SPIS TREŚCI

1

WSTĘP

Rozwój inteligentnego budownictwa w ostatnich latach pozwala przewidywać, że popularność dziedziny będzie rosła z roku na rok. Możliwości konfiguracyjne oraz mnogość dostępnych narzędzi ułatwia implementację coraz to bardziej złożonych systemów automatyki budynkowej. Co więcej, duża część powstających obecnie budynków i obiektów (biurowych, hotelowych, mieszkalnych lub przemysłowych) zawiera w sobie elementy uwzględniające możliwości szybkiej integracji z wykorzystywanymi systemami służącymi do monitorowania i sterowania.

Celem pracy jest opracowanie kompletnego środowiska pozwalającego na budowę autonomicznych systemów diagnostyki, monitorowania, sterowania i zarządzania inteligentnymi budynkami. Częścią środowiska jest dedykowany język (SMOL) opisu funkcji, mechanizmów oraz urządzeń sieciowych. Semantyka języka umożliwić ma precyzyjne odwzorowanie relacji występujących w rzeczywistym systemie sieciowym.

Tezę niniejszej pracy autor formułuje w następującym stwierdzeniu: **Na obecnym etapie postępu technologicznego inteligentne nadzorowanie obiektów złożonych - w tym zarządzanie inteligentnymi budynkami BMS - oprzeć można na projekcie uniwersalnego rozproszonego systemu komputerowego (Sieciowego Monitora Obiektu)**.

Podczas pracy na rozprawą autor posiłkował się przedstawionymi poniżej tezami pomocniczymi:

I teza pomocnicza: Projekt rozproszonego systemu nadzoru i diagnostyki SMO (Sieciowego Monitora Obiektu), będącego uniwersalnym narzędziem do monitorowania i przetwarzania danych pomiarowych, może być oparty na stowarzyszonym języku modelowania (SMOL) oraz - w zakresie implementacyjnym - na architekturze zorientowanej serwisowo (SOA).

II teza pomocnicza: Język modelowania (SMOL) powinien wyrażać wszystkie niezbędne funkcje systemu SMO, opierać się na generycznych strukturach przetwarzania hierarchicznego oraz spełniać podstawowe założenia paradygmatu programowania obiektowego (abstrakcja, hermetyzacja, polimorfizm, dziedziczenie).

Potrzeba stworzenia warunków zbliżonych do rzeczywistych pozwalających na weryfikację działania układów sterownia i diagnostyki była główną motywacją do opracowania środowiska symulacyjnego SMOL-Sim. W wyniku prowadzonych prac badawczych otrzymano narzędzie, które pozwala na sprawdzanie możliwości działania zaprojektowanej konfiguracji sprzętowej w ściśle określonych warunkach, reprezentowanych przez stosowne kryteria. Zadaniem symulatora jest weryfikacja zasad działania platformy SMO w różnych konfiguracjach sieciowych. Platforma pozwala na weryfikację reakcji systemu na sygnały sterujące wysyłane z urządzeń sieciowych. Opracowane rozwiązanie symulacyjne, pozwala na modelowanie i symulację sieci, bądź też jej fragmentów. Finalnym efektem przeprowadzonych prac projektowych i implementacyjnych jest narzędzie pozwalające przeprowadzać weryfikację realizowalności i wykonalności projektowanych rozwiązań pomiarowo-diagnostyczno-sterujących.

Częścią środowiska jest dedykowany język (SMOL) opisu funkcji, mechanizmów oraz urządzeń sieciowych. Semantyka języka umożliwia precyzyjne odwzorowanie relacji występujących w rzeczywistym systemie sieciowym. W zakres realizowanej pracy weszło opracowanie parsera tłumaczącego kodu programu napisanego w języku SMOL na język zrozumiały dla środowiska komputerowego. Język SMOL ma ułatwiać korzystanie z dwóch systemowych bibliotek: biblioteki pozwalającej na precyzyjne odwzorowanie struktury sieciowej (SMOH) oraz biblioteki zawierającej zbiór funkcji Sieciowego Monitora Obiektu (SMOF). Zadaniem projektanta systemu SMO jest zdefiniowanie wymagań (strukturalnych i funkcjonalnych) stawianych przed budowaną siecią.

Proces parsowania służyć ma automatycznej weryfikacji wzajemnej zgodności (realizowalności) elementów SMOH i SMOF oraz pozwalać na budowę schematu struktury sieci. Powinien też umożliwiać zdefiniowanie niezbędnych systemowo uwarunkowań, wraz z elementami optymalizacji strukturalnej, z uwzględnieniem zadanych ograniczeń.

Zwieńczeniem projektu jest opracowanie i implementacja kompletnej platformy symulacyjnej służącej do przeprowadzania testów w celu weryfikacji poprawności zaprojektowanej koncepcji systemu SMO w aspekcie zadań diagnostyki, monitorowania i sterowania. Dysponując zweryfikowanym syntaktycznie (analiza składniowa) i semantycznie (analiza logiczna) opisem w języku SMOL oraz produktem parsowania w postaci graficznej struktury sieci, projektant może przeprowadzić wydajnościowe (obciążeniowe) testy w stowarzyszonym symulatorze-środowisku.

2

SYSTEMY ZARZĄDZANIA W OBIEKTACH INTELIGENTNYCH

Zawartość rozdziału prezentuje gruntowny przegląd aktualnej wiedzy na temat systemów należących do grupy BMS (ang. *Building Management Systems*), korzystając z podziału używanych obecnie podsystemów na systemy zarządzania energią (BEMS), systemy ogrzewania, wentylacji i klimatyzacji (HVAC) oraz systemy zarządzania oświetleniem (BLS). Rozdział stanowi też wprowadzenie do diagnostyki rozłożonych obiektów przemysłowych. Przedstawia się w nim aktualnie wykorzystywane mechanizmy diagnozowania, sposoby agregacji danych pomiarowych (oparte na przeglądzie baz danych, pozwalających na zapis i szybkie wyszukiwanie informacji), a w konsekwencji metody wykorzystywania tychże danych do budowy systemów regułowych (ekspertowych). W materiale tym zawierają się również informacje na temat spopularyzowanych obecnie języków domenowych (DSL) oraz ich potencjalnego wykorzystania w automatyce. Analizie poddawany jest również aktualny stan wiedzy na temat rozproszonych systemów obliczeniowych oraz sposobów ich wykorzystania w diagnostyce i analizie.

2.1. Wprowadzenie

W ostatniej dekadzie jesteśmy świadkami niezwykle dynamicznego rozwoju technologii, który przekłada się na większość dziedzin naszego życie. Motoryzacja, przemysł oraz automatyzacja produkcji to charakterystyczne przykłady, które można by wymienić. Pochodną technologicznej rewolucji jest również wyodrębnienie się nowych dziedzin nauki

skoncentrowanych na badaniu, projektowaniu i wdrażaniu nowoczesnych systemów. Dyrektywa Parlamentu Europejskiego i Rady Europy nr 2010/31/UE zwraca uwagę na fakt, że za funkcjonowanie budynków odpowiada 40% łącznego zużycia energii w Unii Europejskiej, a obserwowany ciągły przyrost infrastruktury budowlanej przyczynia się do wzrostu zużycia energii, która powinna być w coraz większym stopniu pozyskiwana ze źródeł odnawialnych [79]. Rozwój inteligentnej automatyki budynkowej jest odpowiedzą na obecnie rosnące zapotrzebowanie na budowę nowoczesnych w pełni zautomatyzowanych miejsc życia, pracy i wypoczynku.

Rys. 2.1. Inteligentny budynek zaprojektowany przez polskiego projektanta [33].

Bardzo modnym ostatnio określeniem jest pojęcie „inteligentnego budynku", które kojarzymy z obiektem, który w sposób celowy i właściwy, samoistnie reaguje na występujące w jego otoczeniu zdarzenia i zmieniające się czynniki zewnętrzne [88].

Systemy automatyki budynkowej nie są już jedynie rozwiązaniami ułatwiającymi korzystanie i zarządzanie obiektami biurowymi, mieszkalnymi i przemysłowymi. To zbiór zintegrowanych i komunikujących się ze sobą technologii pozwalających nie tylko na zarządzanie, ale również na dążenie do optymalnego wykorzystania zasobów (prądu, gazu czy wody). Dążenie do optymalizacji kosztowej w kontekście eksploatacji budynku przekłada się na coraz ciekawsze rozwiązania pozwalające na osiąganie takiego celu. Jednak wysoka sprawność zarządzania budynkiem może być osiągnięta jedynie wtedy, gdy spojrzymy na obiekt jako na złożony, zintegrowany system, który do utrzymania najwyż-

szych poziomów wydajności operacyjnej wymaga proaktywnego i ciągłego zarządzania oraz konserwacji [34].

Powstają coraz to nowsze systemy automatyki rozproszonej, stanowiące wyposażenie inteligentnego budynku. Rozwój automatyki budynkowej oraz metod poszukiwania optymalizacji w funkcjonowaniu dużych obiektów przekłada się też na powstawanie publikacji naukowych. Szczególnie dynamiczny wzrost w tej dziedzinie widoczny jest na przełomie ostatnich 10 lat. Obfituje on w wieloma książkami, dysertacjami doktorskimi i naukowymi dotykającymi problematyki związanej z integracją automatyki z nowoczesnym budownictwem [97]. W naturalny sposób pojawiają się tu rozwiązania automatyzujące i kontrolujące działanie mieszkań i domów jednorodzinnych. Możliwości konfiguracyjne i instalacyjne sprawiają, że systemy te stają się dostępne dla każdego. Jednym z przykładów jest system Apple Home. Należy podkreślić, że rozwój inteligentnego budownictwa nie jest związany jedynie z automatyką, gdyż mamy tu do czynienia z postępem w budownictwie, technologii materiałowej oraz nowoczesnych systemach grzewczo-wentylacyjnych. Potrafimy obecnie budować domy w miejscach, w których wcześniej byłby to całkowicie niemożliwe. Nauczyliśmy się wykorzystywać źródła geo-termalne do optymalnego ogrzewania. Podobnie zastosowanie paneli słonecznych oraz nowoczesnych systemów magazynowania energii przyczynia się do zmniejszenia zapotrzebowania lub zwiększenia niezależności energetycznej [**?**]. Jednym z przykładów nowoczesnych rozwiązań budowlanych jest dom zaprojektowany przez polskiego architekta Roberta Koniecznego, który został wybrany najlepszym budynkiem świata w konkursie Wallpaper Design Awards 2017 [33]. Zaprojektowany budynek przedstawiono na rys 2.1. Synergia pomiędzy adekwatnymi rozwiązaniami powinna mieć bezpośredni wpływ na produkt końcowy, jakim jest inteligentny budynek.

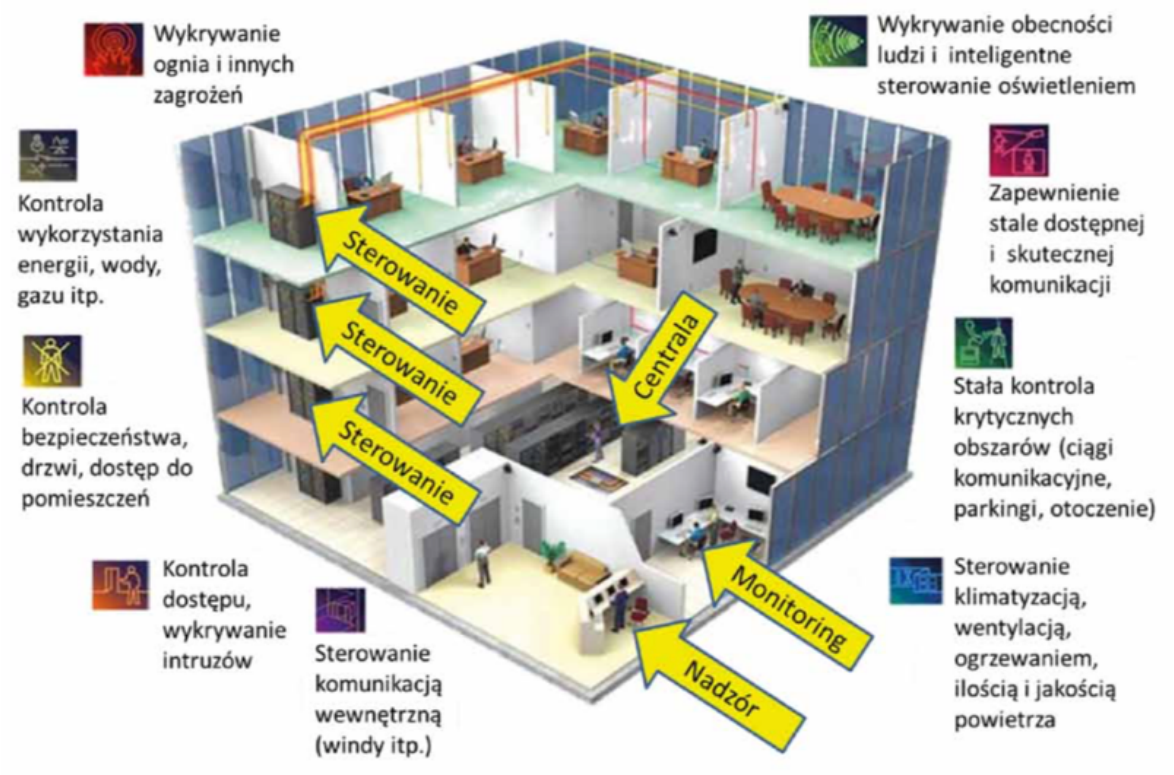
Oszczędności po zastosowaniu tej technologii są na tyle znaczące, że równoważą koszty poniesione na wdrożenie. Co stanowi niewątpliwą motywację do poszukiwania jeszcze lepszych sposobów wykorzystania automatyki w tym zakresie.

Reasumując, inteligentne budownictwo jest dziedziną multidyscyplinarną, która łączy wiedzę podchodzącą z różnych obszarów techniki. Efektem realizowanej synergii są rozmaite nowoczesne rozwiązania mieszkalne, użytkowe lub przemysłowe wybiegające daleko poza podstawową definicję słowa budynek. W niniejszym zaś rozdziale dokonamy przeglądu technologii, metod oraz systemów związanych z inteligentnym budownictwem.

2.2. Systemy sterowania w inteligentnych budynkach

W celu skonkretyzowania analizie poddamy model inteligentnego budynku, którego schemat modelu na rys. 2.2. Na rysunku zaznaczono w sposób poglądowy systemy dostępne w tego typu rozwiązaniach. W dalszej części pracy przedstawione i scharakteryzowane

zostaną wyszególnione podsystemy oraz sposoby pozwalające na ich skuteczną integrację.



Rys. 2.2. Główne systemy oraz podsystemy stosowane w inteligentnych budynkach [97].

2.2.1. Podsystemy inteligentnego budynku

W inteligentnym budynku stosujemy przedstawiony poniżej podział systemów i podsystemów automatyki budynkowej, które mają zapewnić bezpieczeństwo obiektu na najwyższym poziomie. System automatyki budynkowej (BAS[1]) stanowi rozwiązanie pozwalające na integrację oraz optymalne wykorzystanie poszczególnych podsystemów.

2.2.2. System sygnalizacji włamania i napadu oraz system telewizji przemysłowej

W systemie sygnalizacji włamania i napadu (SSWiN) wyróżnia się infrastrukturę oraz czujniki. W infrastrukturze mieści się centrala i manipulatory oraz wszelkie urządzenia

[1]ang. *Building Automation System*.

wyjściowe: sygnalizatory, urządzenia monitoringu i powiadamiania (GSM, telefon, radio, e-mail), jak również urządzenia wykonawcze, w tym rejestratory, których zadaniem jest raportowanie sytuacji alarmowych. Czujki, dostarczają do centrali systemu informacji o naruszeniu strefy bezpieczeństwa, wystąpieniu sytuacji alarmowej lub przekroczeniu wartości ekstremalnej. Do typowych czujników należą:

- czujki ruchu, głównie sensory pasywne podczerwieni (PIR[2]), mikrofalowe (MW[3]) oraz ultradźwiękowe (US[4]), jak również elementy dualne wykorzystujące dwa detektory z wyżej wymienionych - kryterium alarmu dla tych czujek jest zmiana sygnału docierającego do detektora w wyniku przemieszczania się intruza generującego promieniowanie IR[5] (różne od promieniowania tła, PIR) lub w skutek zjawiska Dopplera (US i MW)
- aktywne czujniki (bariery), złożone z nadajnika i odbiornika, dla których kryterium alarmu jest przerwa w kontakcie optycznym pomiędzy nadajnikiem a odbiornikiem
- czujka stłuczenia szyby - mikrofonowe urządzenie reagujące na częstotliwości fal charakterystyczne dla uderzenia w taflę szklaną oraz dla tłuczenia szkła
- czujki udarowe (wibracyjne) reagujące na wstrząsy, będące następstwem uderzenia
- czujki otwarcia drzwi i okien (najczęściej magnetyczne/kontaktronowe lub elektromechaniczne/stykowe)
- czujki przekroczenia progowego stężenia gazu (wodoru, dwutlenku węgla, metanu, butanu czy par chloroformu)
- optyczne czujki dymu
- czujki przekroczenia progowego poziomu cieczy.

Zazwyczaj w jednym obiekcie instalowane są przynajmniej dwa systemy ochronne, wspomniany wcześniej system SSWiN oraz zintegrowany z nim system telewizji przemysłowej (CCTV[6]). Co wynika przede wszystkim ze stosunkowo dużej kompatybilności pomiędzy tymi urządzeniami.

Wyjścia alarmowe czujek są zwykle przystosowane do podłączenia typowych urządzeń wyposażonych w wejścia stykowe typu NO[7] lub NC[8], jakimi są urządzenia SSWN

[2]ang. *Passive Infrared*.
[3]ang. *Microwave*.
[4]ang. *Ultrasonic*.
[5]ang. *Infrared*.
[6]ang. *Closed-Circuit Television*.
[7]ang. *Normally Open*.
[8]ang. *Normally Closed*.

i CCTV. W takim przypadku połączenie jest bezpośrednie bez dodatkowych adapterów czy konwerterów.

Najczęściej przy połączeniu systemów ochronnych można skorzystać z wyjść i wejść alarmowych zlokalizowanych w centrali SSWN (moduł rozszerzeń lub moduł wyjść programowalnych) oraz w rejestratorze lub multiplekserze CCTV. Dzięki temu powstaje swoiste sprzężenie zwrotne:

- w momencie wykrycia ruchu lub innego zjawiska stanowiącego kryterium alarmu przez czujki następuje podanie sygnału przez jedno z wyjść programowalnych SSWN na wejście alarmowe CCTV, a w konsekwencji uaktywnienie zapisu alarmowego na rejestratorze (najczęściej poprzedzone nagraniem w trybie prealarmowym)
- w momencie detekcji ruchu na obrazach rejestrowanych w systemie CCTV następuje uaktywnienie wyjścia alarmowego, co z kolei aktywuje wejście alarmowe SSWN oraz powoduje wygenerowanie odpowiednio zaprogramowanego alarmu (optyczno-akustycznego) wraz z uaktywnieniem toru monitoringu alarmu.

Wobec powyżeszego, każdemu wykryciu zjawiska alarmowego zarówno w strukturze SSWN, jak i w CCTV, zawsze towarzyszy alarm oraz nagranie obrazów z kamer. Funkcja nagrywania prealarmowego pozwala na wykrycie i udokumentowanie zapisem przyczyny alarmu.

Zapis cyfrowy w systemach CCTV stworza dodatkowe możliwości. Na przykład w chwili wykrycia intruza przez czujki SSWN, poza załączeniem rejestracji obrazu i fonii na twardym dysku, możliwe jest też przesłanie alarmowej wiadomości (MMS lub e-mail) pod wcześniej ustalony adres [28].

2.2.3. Systemy ostrzegawcze i rozgłoszeniowe

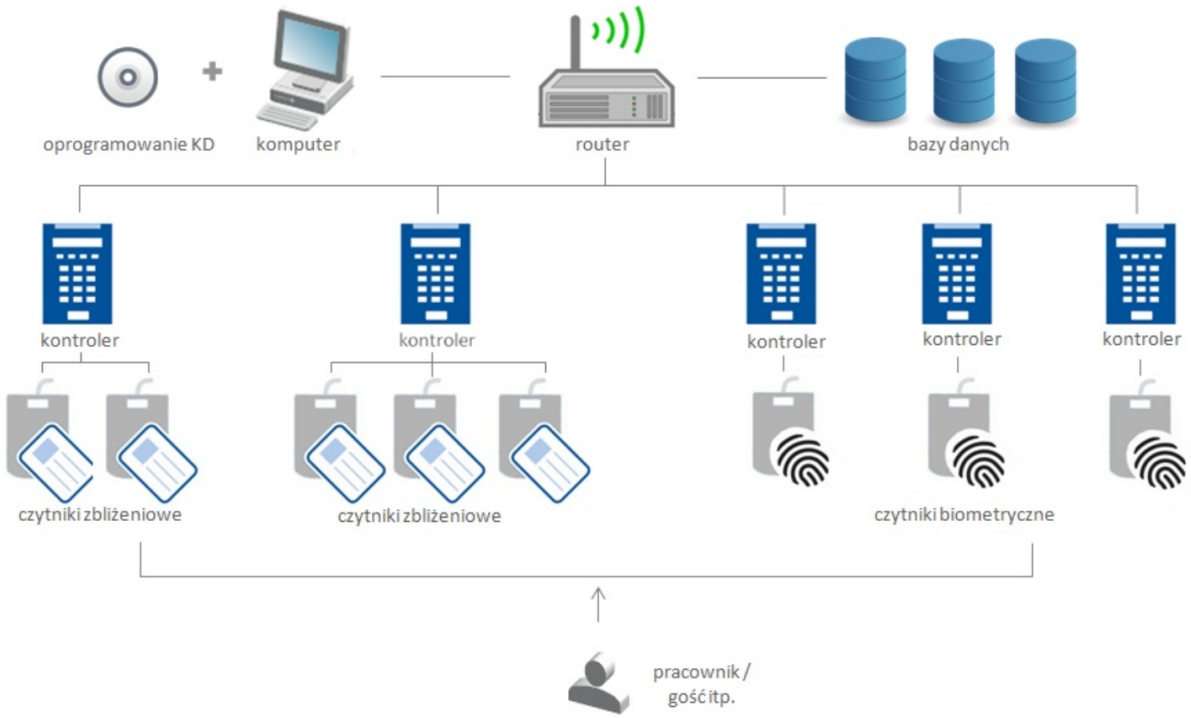
Kolejną grupą systemów występujących w systemach IB są dźwiękowe systemy ostrzegawcze i rozgłoszeniowe (DSO[9], DSR[10]). Służą do szybkiego i uporządkowanego informowania za pomocą komunikatów ostrzegawczych osób znajdujących się wewnątrz zagrożonego obiektu. Rolą tych systemów jest zabezpieczanie życia i mienia, jak również wspomaganie w przeprowadzaniu ewakuacji w przypadku zagrożenia pożarowego. Wykorzystywane są też one do odtwarzania muzyki lub rejestracji wystąpień w salach konferencyjnych lub pomieszczeniach wspólnych. W skład takich systemów wchodzą wzmacniacze dźwięku, miksery i głośniki [35].

[9]DSO - Dźwiękowy System Ostrzegawczy.
[10]DSR - Dźwiękowy System Rozgłoszeniowy.

2.2.4. Systemy kontroli dostępu

Kontrola dostępu stanowi jeden z ważniejszych systemów występujących w inteligentnych budynkach. Pozwala na identyfikację osób lub pojazdów uprawnionych do przekroczenia granicy obszaru zastrzeżonego oraz umożliwienie im wejścia i wyjścia z tego obszaru. Głównym zadaniem systemu kontroli dostępu (oznaczanym z języka angielskiego skrótem AC[11]) jest dokonywanie selekcji oraz uprządkowanie i ograniczenie ruchu osób lub pojazdów na terenie objętym działaniem systemu, jak również archiwizację takich zdarzeń. Systemy kontroli dostępu do określonych stref w inteligentnych budynkach realizują zadanie ograniczenia dostępu do określonych miejsc osobom niepowołanym lub niebezpiecznym. Schemat działania systemu KD[12] przedstawiono na rys. 2.3.

Podstawowymi elementami składowymi każdego systemu KD są elementy takie jak czytniki, kontrolery i oprogramowanie, które poniżej przedstawiono i scharakteryzowano.



Rys. 2.3. Główne systemy oraz podsystemy KD stosowane w inteligentnych budynkach [47].

Czytniki stanowią podstawową grupę urządzeń wchodzących w skład systemów KD. Ich zadaniem jest identyfikacja obiektów (osób lub pojazdów) przekraczających granicę strefy. Do najbardziej popularnych należą czytniki kart magnetycznych, czytniki zbliże-

[11] ang. *Access Control*.
[12] KD - kontrola dostępu.

niowe RFID[13], czytniki kart chipowych oraz sensory biometryczne umożliwiające odczyt linii papilarnych, geometrii dłoni, czy też tęczówki oka. Z grupy wymienionych czytników zdecydowanie największe bezpieczeństwo zapewniają czujniki biometryczne [9]. Znalezienie dwóch osób posiadających jednakowe linie papilarne lub jednakowe tęczówki oka jest praktycznie niemożliwe. Karty zbliżeniowe są najczęściej wykorzystywanymi rozwiązaniami, które pozwalają na kompromis pomiędzy ceną i bezpieczeństwem. Nowa generacja tej technologii łączy wygodę użytkowania z podwyższonym bezpieczeństwem (większa pamięć i dodatkowe funkcje umożliwiające zapis i odczyt). Większość nowoczesnych czytników wyposażona jest w funkcję wykrywania sabotażu, która reaguje uruchomieniem alarmu podczas każdej próby usunięcia czytnika. Nowoczesne systemy KD monitorują dodatkowo czytniki za pomocą kamer z funkcja wideorejestracji. Każdorazowe użycie czytnika wywołuje dodatkową rejestrację wideo. Takie rozwiązanie umożliwia pełną weryfikację osoby, na podstawie porównania zdjęcia przechowywanego w systemie ze zdjęciem z kamery, wykonanym podczas korzystania z czujnika. Istnieją czytniki pozwalające na pełnienie funkcji czytnika wejść i wyjść. W połączeniu z funkcją *antipassback* zapobiega to wejściu nieuprawnionej osoby do strefy dzięki użyciu tej samej karty. Czujnik pozwala na ponowne wejście do strefy dopiero, gdy odnotuje wyjście.

Kontrolery służą do wymiany danych pomiędzy czytnikiem i oprogramowaniem. Stanowi warstwę pośrednią zawierającą wszystkie niezbędne informacje do pracy czytnika. Takie rozwiązania zapobiega sytuacji, w której wejście do strefy nie jest strzeżone w przypadku zaniku połączenia sieciowego pomiędzy serwerem i kontrolerem. W przypadku wyposażenia kontrolera w zasilanie awaryjne, nawet przy zaniku napięcia możliwe będzie dalsze otwieranie drzwi i rejestrowanie osób odblokowujących wejście. Kontrolery pozwalają na integrację z systemami kamer, umożliwiając ich uruchamianie albo przełączanie pomiędzy trybem obserwacji i zapisu.

Oprogramowanie stanowi grupę niezbędną do poprawnego funkcjonowania systemów KD. Jest to warstwa dostępowa, tj. umożliwiająca dostęp do bazy danych, która zawiera następujące informacje:

- dane osobowe
- identyfikatory przypisane poszczególnym osobom (np. numery kart zbliżeniowych)
- prawa dostępu poszczególnych osób do poszczególnych pomieszczeń
- dane dotyczące czujników i strzeżonych stref.

Zarządzaniem systemem z poziomu oprogramowania zajmuje się administrator. Może on dokonywać łączenia osób w grupy, przypisywać im odpowiednie uprawnienia, jak również łączyć czytniki w strefy - zarządzając tym samym obszarami wymagającymi auto-

[13]ang. *Radio-Frequency Identification.*

ryzowanego dostępu.

W nowoczesnych systemach kontroli dostępu, wybór odpowiedniej metody identyfikacji decyduje o niezawodności i koszcie wdrażanego systemu. Wśród dostępnych metod wyróżnić można trzy podstawowe grupy niezawodności.

Do najniższej grupy zalicza się metody oparte na przedmiocie w postaci klucza (karty chipowe, magnetyczne, zbliżeniowe). Słabym aspektem tych metod jest fakt, że autoryzacja może być dokonywana przez niewłaściwą osobę. Klucz może bowiem zostać zgubiony, skradziony lub też udostępniony innej osobie.

Bardziej niezawodne podejście reprezentują metody oparte na kodzie, haśle lub innej procedurze strzegącej dostępu do określonej strefy. W tym przypadku wybór odpowiedniego hasła ma bardzo duże znacznie: zbyt łatwe hasło można pozwolić na jego szybkie złamanie, natomiast zbyt złożone może okazać się zbyt trudne do zapamiętania.

Najsilniejsza grupa opiera się na identyfikacji przez rozpoznanie niepowtarzalnych cech fizycznych. Grupa czytników biometrycznych pozwala na analizę wielu ludzkich cech, takich jak odcisk palca, tęczówka oka (wzór kolorów), siatkówkę (wzór naczyń krwionośnych), głos, dłoń (kształt palców i grubość dłoni), twarz (położenie oczu, nosa i ust) oraz pismo ręczne (dynamika pióra w ręce). To właśnie czujniki biometryczne pozwalają na uzyskanie najlepszego stopnia bezpieczeństwa w oparciu o identyfikację [26]. Skanery laserowe pozwalają na budowanie systemów o najwyższym poziomie bezpieczeństwa oraz najniższej z możliwych liczbie fałszywych alarmów [15].

Jednym z przykładowych urządzeń biometrycznych jest czytnik RFT-1000 opracowany przez polską firmę Roger. Jest to przykład rozwiązania hybrydowego posiadającego wbudowany skaner linii papilarnych oraz czytnik kart zbliżeniowych opracowanych w standardzie ISO/IEC 14443A[14]. Czytnik pozwala na pracę w dwóch trybach rozpoznawania użytkowników. Pierwszy tryb polega na porównaniu zeskanowanego odcisku palca z wzorcami przechowywanymi w wewnętrznej bazie danych czytnika (tzw. tryb 1:N) lub z wzorem odcisku palca wczytanym z karty zbliżeniowej (tzw. tryb 1:1). Czytnik zapewnia możliwość zarejestrowania do 1900 wzorów linii papilarnych w wewnętrznej pamięci. Dodatkową cechą opisywanego urządzenia jest wykorzystanie standardu AES128 CBC[15] do komunikacji z czujnikiem. Takie rozwiązania zwiększa odporność urządzenia na potencjalne ataki cybernetyczne [36]. Wszystkie opisane wyżej grupy urządzeń umożliwiających identyfikację można ze sobą łączyć, tworząc rozwiązania hybrydowe powiększające bezpieczeństwo kontrolowanych stref.

[14]ISO/IEC 14443A - https://pl.wikipedia.org/wiki/ISO/IEC_14443.
[15]AES - https://en.wikipedia.org/wiki/Advanced_Encryption_Standard

2.2.5. Systemy domofonowe i wideofonowe

Kolejną grupą podsystemów występujących w rozwiązaniach takich, jak inteligentne budynki, są urządzenia zapewniające komunikację głosową (wideo) na małych odległościach. Urządzenie te nazywane są domofonami lub wideofonami. Komunikacja w tym rozwiązaniu odbywa się pomiędzy kasetą bramową a słuchawką montowaną wewnątrz zwaną unifonem. W przypadku domofonów wyróżniamy dwie podstawowe grupy, urządzenia analogowe i cyfrowe. Systemy analogowe stanowią dominującą grupę na rynku i najczęściej wykorzystywane są w budownictwie jednorodzinnym. W takich realizacjach wykorzystywany jest przeważnie jeden bramofon i jeden lub dwa unifony (każdy bramofon jest połączony z unifonem za pomocą oddzielnego kabla). Konstrukcja typowego domofonu nie jest skomplikowana, opiera się bowiem na wzmacniaczu akustycznym z tranzystorem i układem scalonym oraz dołączonymi do wejścia i wyjścia głośnikami dynamicznymi.

Zdecydowanie bardziej zaawansowanymi rozwiązanymi są urządzenia w wersji cyfrowej. Technologia ta sprawdza się szczególnie dobrze w domach wielorodzinnych, korzystających z centralki cyfrowej z dekoderami. W centrali zlokalizowana jest klawiatura numeryczna, za pomocą, której można wybrać odpowiedni numer lokalu lub wprowadzić kod dostępu, aby otworzyć drzwi lub furtkę. W przypadku rozwiązań cyfrowych stosukowo łatwa jest integracja pomiędzy unifonem a systemem ochrony budynku, jak również systemami kontroli dostępu (KD). W takich rozwiązaniach drzwi mogą być otwierane z wykorzystaniem kodu, karty zbliżeniowej lub akceptacji osoby trzeciej. Bardziej zaawansowane urządzenia pozwalają na sterowanie kilkoma obwodami wykonawczymi jednocześnie np. na otwieranie dodatkowych drzwi lub sterowania bramą.

Wideofony są najbardziej rozbudowanymi urządzeniami dostępowymi, które wyposażone są w kolorowe kamery i ciekłokrystaliczne monitory. Kamera uruchamiana jest w momencie naciśnięcia przycisku w bramofonie. Jeśli wideofon nie zostanie odebrany (np. brak osoby w budynku), kamera jest uruchamiana i rozpoczyna się rejestracja obrazu. Nowoczesne wideofony pozwalają również na monitoring otoczenia. W takiej konfiguracji kamera rejestruje obraz automatycznie, zapisując go w określonych odstępach czasu. Dzięki zewnętrznej karcie istnieje możliwość przeniesienia nagranych informacji do komputera [31] . W rozwiązaniach przemysłowych istnieje możliwość budowy rozwiązania łączącego kamery CCTV oraz urządzenia domofonowe. Taki system daje zdecydowanie większe możliwości weryfikacji osoby (na podstawie obrazu z kilku kamer) przed udzieleniem dostępu do określonej strefy lub pomieszczenia [32].

2.2.6. Systemy przeciwpożarowe

Jednym z największych zagrożeń dla bezpieczeństwa ludzi w obiekcie jest pożar, dlatego też grupa systemów przeciwpożarowych (PP) jest kolejną z grup systemów instalowanych

i wykorzystywanych w inteligentnych obiektach. Spośród wszystkich dostępnych systemów IB system PP powinien być nadrzędnym, ponieważ jego kluczowym zadaniem jest zagwarantowanie bezpieczeństwa przebywającym w budynku ludziom.

Systemy odpowiedzialne za ochronę przeciwpożarową w inteligentnym budynku dzielą się na dwie grupy: systemy czynnej ochrony przeciwpożarowej oraz systemy użytkowe (ważne z punktu widzenia bezpieczeństwa pożarowego). Do pierwszej grupy należą: system sygnalizacji pożarowej, dźwiękowy system ostrzegawczy, system oddymiania, stałe urządzenia gaśnicze, system sterowania elementami oddzieleń pożarowych (klapy pożarowe, bramy pożarowe) oraz wewnętrzne sieci hydrantowe. Do drugiej grupy kwalifikowaną się: system wentylacji, system klimatyzacji, system sterowania windami, system oświetlenia ewakuacyjnego i awaryjnego oraz system kontroli dostępu.

Najważniejszą rolę wśród systemów odpowiedzialnych za ochronę przeciwpożarową w obiekcie pełni system sygnalizacji pożarowej (SSP). Głównym jego zadaniem jest wczesne wykrycie pożaru i wysterowanie systemów z nim współpracujących, tak aby zapewnić bezpieczną i możliwie jak najszybszą ewakuację ludzi z budynku oraz zapobiec rozprzestrzenianiu się pożaru (minimalizacja skutków).

Uzupełnieniem systemu PP poprawiającym obsługę i eksploatację jest zastosowanie komputerowego systemu wizualizacji i zarządzania. Rozwiązania takie stosuje się w celu ułatwienia obsługi systemu sygnalizacji pożarowej oraz oceny poprawności działania systemu w przypadku wystąpienia alarmu pożarowego [78].

2.2.7. Systemy sterowania oświetleniem

Grupą systemów, bez których działanie inteligentnego budynków byłoby praktycznie niemożliwe, są automatyczne systemy sterowania oświetleniem. Zapewniają one nie tylko odpowiednie warunki świetlne w określonych pomieszczeniach i strefach budynku, ale również racjonalnie niskie zużycie energii przekładające się w sposób bezpośredni na ograniczenie kosztów eksploatacji (oświetlenie typowego budynku użytkowego generuje około 40% rocznych kosztów energii). Zaawansowane sterowanie oświetleniem wbudowane w nowoczesne systemy monitoringu i sterowania pozwalają na efektywne zarządzanie oświetleniem całego budynku z jednego centralnego miejsca, co usprawnia również konserwację i eksploatację.

Podstawową funkcją tej grupy systemów jest zapewnienie optymalnych parametrów świetlnych w określonych miejscach budynku poprzez dobór odpowiednich strategii sterowania oświetleniem. Sześć podstawowych strategii wykorzystywanych w różnych konfiguracjach systemów oświetleniowych scharakteryzowano poniżej.

Czujniki ruchu/obecności – w tej strategii czujniki odpowiedzialne są za detekcję osób przemieszczających się lub znajdujących się w poszczególnych pomieszczeniach. Umożliwia to automatyczne włączanie, podtrzymywanie i - przy braku obecności - wy-

łączanie oświetlenia. Czujniki mogę działać w dwóch trybach: autonomicznym oraz sieciowym. Tryb autonomiczny pozwala czujnikowi na bezpośrednie załączenie oświetlenie, natomiast sieciowy przekazuje sygnał z czujnika do odpowiedniego sterownika zarządzającego oświetleniem. Czujniki działają w technologii pasywnej poczerwieni PIR (reagują na połączenie ruchu i ciepła wytwarzanego przez użytkowników) generując sygnały załączenia i wyłączenia światła w sposób automatyczny. Rozwiązanie to doskonale sprawdza się w zastosowaniu do różnego rodzaju budynków i przestrzeni. Przykładem są inteligentne obiekty biurowe, w których znajduje się wiele obszarów o różnym natężeniu ruchu np. korytarze, pomieszczenia techniczne, garaże, itp. Drugą grupą czujników są urządzenia zbudowane w oparciu o technologię radarową. W stosunku do PIR, czujniki radarowe wykrywają każdy rodzaj ruchu, nie rozróżniając obiektów generujących ciepło. Obie technologie mogą ze sobą współpracować i wzajemnie się uzupełniać.

Wykorzystanie światła dziennego – też może być stosowana do sterowania oświetleniem w ten sposób, że natężenie światła elektrycznego automatycznie dostosowuje się do ilości światła dziennego dostępnego w danym pomieszczeniu. Nowoczesne systemy sterowania oświetleniem działają zatem adaptacyjnie w stosunku do światła dziennego, które cechuje się dużą zmiennością intensywności oświetlenia oraz składu widmowego w zależności od pory dnia i warunków pogodowych.

Harmonogramowanie czasowe – pozwala na opracowanie scenariusza zarządzania oświetleniem w określonych obszarach na podstawie z góry określnego i konfigurowalnego harmonogramu.

Strojenie zadaniowe lub sterowanie indywidualne – kiedy maksymalne poziomy natężenia oświetlenia ustawiane są z góry i przypisywane do konkretnych zadań lub pomieszczeń (zapobieganie zbyt mocnemu oświetleniu określonych obszarów). W przypadku sterowania indywidualnego, kiedy to użytkownik dostosowuje natężenie oświetlenie do własnych potrzeb poprzez wykorzystanie graficznego interfejsu umieszczono na panelach dotykowych lub tabletach.

Odpowiedz na zapotrzebowanie – scenariusz polegający na zmniejszeniu obciążenia sieci przez automatycznie przyciemniane lub wyłączanie w ten sposób oświetlenia prowadzi do zmniejszenia zapotrzebowania na energię całego budynku lub oszczędności w okresach podwyższonego zapotrzebowania na energię.

Systemy oświetleniowe pozwalają na implementację wszystkich sześciu opisanych scenariuszy jednocześnie. Efektywność działania oraz generowane oszczędności wynikające z ich wdrożenia przyczyniają się do kontynuacji prac nad doskonaleniem tych systemów.

W przypadku systemów oświetleniowych, efektywność ich działania zależna jest od protokołów wykorzystywanych w komunikacji pomiędzy czujnikami a urządzeniami sterującymi. W dalszej części scharakteryzowane zostaną protokoły najczęściej wykorzystywane w inteligentnym sterowaniu oświetleniem.

System DALI[16] – prezentowany w 2000 roku na międzynarodowych targach we Frankfurcie, stanowi standard interfejsu (open-source) dla elektronicznych układów zasilających z możliwością regulacji strumienia świetlnego. Sieć DALI złożona jest z jednego kontrolera i wielu urządzeń typu *slave*. Urządzenie kontrolujące, poprzez wykorzystanie adresowania (unikalnej liczby w przedziale od 0 do 63) może nawiązywać dwukierunkową komunikację z innymi urządzeniami. W sieci, które posiadają więcej niż 64 urządzenia, stosuje się tzw. bramki DALI, aby rozszerzyć liczbę obsługiwanych urządzeń. Komunikacja pomiędzy kontrolerem a urządzeniami odbywa się asynchronicznie według protokołu szeregowego w trybie *half-duplex*. Medium komunikacyjnym jest kabel dwużyłowy, pozwalający na osiąganie transferu do 1200 bit/s [41]. Sieci można budować z wykorzystania topologii szyny lub gwiazdy. Otwartość kodu, łatwość w konfiguracji oraz funkcjonalność powodują, że rozwiązanie to jest chętnie wybierane i stosowane w zarówno w przypadku pojedynczych pomieszczeń, jak i złożonych koncepcji oświetleniowych dla całych pięter lub obiektów wielopiętrowych. Niewątpliwie dużą zaletą systemu DALI jest łatwość jego integracji z istniejącymi systemami automatyki budynku oraz fakt, że system ten opracowany został na potrzeby sterowania jedynie oświetleniem.

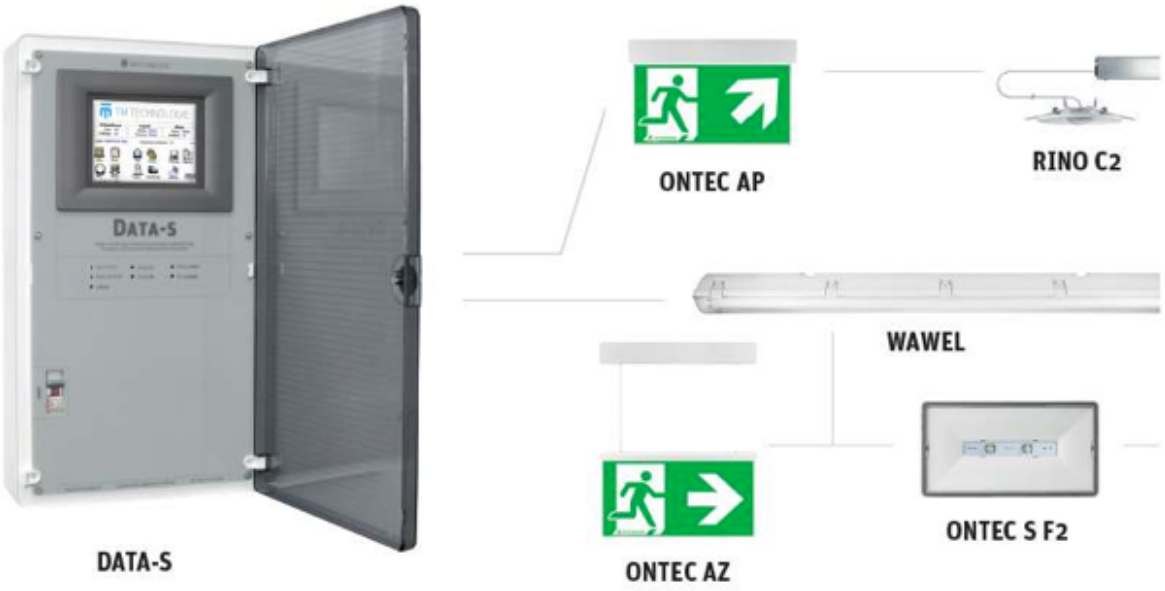
System KNX[17] (EN 50090, ISO/IEC 14543) – też pozwala na sterowanie oświetleniem, jednak jego możliwości znacznie wykraczają poza tę funkcję. KNX umożliwia integrację różnych funkcji automatyki budynkowej, niezależnie od producenta sprzętu. Ze względu na duże możliwości integracyjne system umożliwia sterowanie ogrzewaniem, roletami, markizami, bramami, oknami połaciowymi oraz systemami bezpieczeństwa. KNX stanowi jednolity otwarty standard często wybieranym do integracji systemów w inteligentnych obiektach. Dokładny opis tego standardu znajduje się w punkcie 2.2.9.

System DMX[18] – reprezentuje cyfrowy protokół komunikacyjny wykorzystywany powszechnie do sterowania oświetleniem na scenach teatralnych, w muzeach itp. System ten bazuje na wykorzystaniu interfejsu szeregowego RS-485[19], który służy do sterowania oświetleniem wielokolorowym, RGB[20] oraz temperaturą oświetlenia (np. w teatrze). Za pomocą systemu możliwe jest tworzenie efektów świetlnych z wykorzystaniem się technologii LED[21] (odznaczającej się dużym bogactwem kolorów oraz wydajnością). DMX umożliwia sterowanie maksymalnie 512 kanałami jednocześnie, przy prędkości transmisji danych wynoszącej 250 kB/s (takie parametry systemu pozwalają na sterowanie dużą liczbą lamp RGB przy dynamicznej zmianie barw).

System LonWorks[22] – wyraża standard (ISO/IEC 14908) opracowany przez firmę

[16] ang. *Digital Addressable Lighting Interface.*
[17] KNX - https://en.wikipedia.org/wiki/KNX_(standard).
[18] ang. *Digital Multiplex.*
[19] RS-485 - interfejs szeregowy.
[20] ang. *Red, Green and Blue (color model).*
[21] ang. *Light Emitting diode.*
[22] ang. *Local Operating Network.*

Echelon[23] do budowy rozproszonych systemów automatyki i sterowania. Komunikacja pomiędzy urządzeniami pochodzącymi od różnych producentów realizowana jest z wykorzystaniem otwartego protokołu LonTalk. Wyróżnia się tu pomiędzy czujnikami, elementy wykonawcze i sterowniki. Wymiana danych realizowana jest zdarzeniowo [38]. Rozwiązanie to ze względu na duże możliwości konfiguracyjne sprawdza się dobrze nie tylko w przypadku sterownia oświetleniem, ale również w innych rozwiązaniach dotyczących automatyki budynkowej, np. w kontrola dostępu (2.2.4), sygnalizacji przeciwpożarowej (2.2.6) oraz systemach HVAC (2.2.8).



Rys. 2.4. System monitoringu opraw oświetlenia awaryjnego [44].

System EnOcean to technologia radiowa, niewymagająca używania baterii do zasilania urządzeń. Łączniki i czujniki pobierają energię z otoczenia; w przypadku przełącznika jest to siła potrzebna do jego uruchomienia, w przypadku czujników jest to ciepło lub energia świetlna. Każdy z występujących w sieci nadajników posiada unikalny numer, dzięki któremu umożliwia jest identyfikacja urządzenia po stronie odbiornika. Transmisja danych (mimo ograniczeń energetycznych) została zabezpieczona i uodporniona na zakłócenia. EnOcean jest technologią pozwalającą na duża elastyczność w trakcie projektowania przy stosunkowo niskich kosztach instalacji [37, 83]. Technologia ta pozwana nie tylko na budowanie inteligentnych systemów oświetleniowych, ale również podobnie jak LonWorks innych rozwiązań wchodzących w skład automatyki budynkowej [2].

Oświetlenie ostrzegawcze, niezbędne w systemach zarządzających inteligentnymi obiektami, również nadzoruje za pomocą dedykowanych systemów oświetleniowych. Często w

[23]Echelon – http://www.echelon.com/.

integruje się systemy oświetleniowe z systemami sygnalizacji przeciwpożarowej (rozdział 2.2.6). Do obsługi oraz wizualizacji działania takich systemów wykorzystuje się ekrany dotykowe wykonane w technologii rezystancyjnej lub pojemnościowej. W systemach zarządzających oświetleniem ostrzegawczym istnieje możliwość identyfikacji i oprogramowania każdego urządzenia z osobna (każda lampa posiada inny adres). Przykładowy schemat działania systemu przedstawiony został na rys. 2.4 [102].

2.2.8. Systemy HVAC

Jedną z bardzo zaawansowanych grup systemów pozwalających na wykorzystanie automatyki budynkowej są zintegrowane systemy do sterowania ogrzewaniem, klimatyzacją i wentylacją HVAC[24]. Jest to grupa systemów odgrywająca kluczową rolę przy zapewnianiu wymuszonych parametrów środowiskowych osobom przebywającym w budynku. Zadaniem takiego systemu jest również zapewnienie odpowiednich warunków w pomieszczeniach laboratoryjnych lub wystawowych (np. w muzeach). Systemy HVAC należą niestety do rozwiązań generujących najwyższe koszty związane z utrzymaniem budynku. Dążenie do minimalizacji kosztów eksploatacji systemów wymaga odpowiedniej konfiguracją urządzeń składowych. Zalecane rozwiązanie stanową rozproszone systemy BMS[25], ułatwiające integrację (ogrzewania, klimatyzacji, wentylacji) z innymi podsystemami infrastruktury budynkowej.

Systemy HVAC poprawnie działają w przypadku właściwej regulacji, opartej na bieżącym odczycie wielkości fizycznej (regulowanej) oraz porównaniu jej z wielkością zadaną. W konsekwencji wywierany jest pożądany wpływ na obiekt regulacji. Nawet przy oddziaływaniu zakłóceń, różnica pomiędzy tymi wielkościami utrzymywana jest w założonym zakresie. W obiektach przemysłowych, mieszkalnych oraz biurowych wielkościami zakłócającymi mogą być m.in.: czynniki pogodowe (np. temperatura zewnętrzna, wiatr) oraz zmiany obciążeń cieplno-wilgotnościowych pomieszczeń (następstwo obecności ludzi, otwierania drzwi i okien). Do poprawnego działania system wymaga odpowiedniej konfiguracji urządzeń systemu automatycznej regulacji, składającego się z części pomiarowej (czujniki: temperatury, wilgotności, jakości powietrza itp.), centralnej (elektronicznej), bądź cyfrowej (sterowniki) oraz wykonawczej (zawory regulacyjne). Uzyskiwanie dużej efektywności działania systemów HVAC uzależnione jest w znacznej mierze od skoordynowania urządzeń, tworzących jeden wspólny system sterowania i wzajemnego oddziaływania. Zasadniczą rolą w przypadku rozproszonych systemów regulacji jest zapewnienie możliwości bezpiecznej transmisji danych, odpornej na zakłócenia. Na przestrzeni ostatnich lat wyodrębniło się kilka standardowych protokołów powszechnie wykorzystywanych inżynierii automatyki budynkowej. Do tej grupy zaliczamy rozwiązania takie jak

[24]ang. *Heat Ventilation Air Condition*.
[25]ang. *Building Management System*.

np. LonWorks, BACNet i inne. W dalszej części pracy protokoły zostaną przedstawione i szczegółowo scharakteryzowane.

Wykorzystanie protokołu komunikacyjnego pozwala na sprawną wymianę informacji pomiędzy programowalnymi sterownikami koordynującymi na bieżąco pracę urządzeń HVAC, pomp ciepła, centrali klimatyzacyjnych, wentylatorów wewnętrznych, itd. Nowoczesne systemy sterowania zbierają i zapisują informacje o działaniu systemu, informując jednocześnie użytkownika o wszelkich nieprawidłowościach. Funkcje wykonawcze w systemach HVAC realizują regulatory bezpośredniego działania, które używane są do regulacji (bądź redukcji) temperatury (termostaty przygrzejnikowe, regulatory temperatury ciepłej wody, ograniczniki temperatury powrotu), ciśnienia, różnicy ciśnień, przepływu, poziomu, itd. Wykonywane są one także jako wielofunkcyjne regulatory bezpośredniego działania, służąc np. do jednoczesnej regulacji różnicy ciśnień i przepływu wody sieciowej w węźle. Przykład regulatora wielofunkcyjnego przedstawiono na rys. 2.5.

Rys. 2.5. Regulator wielofunkcyjny RRV934 firmy SIEMENS [29].

Zastosowanie w systemach HVAC znajdują również sterowniki cyfrowe zwane również regulatorami DDC[26]. Działają one w dziedzinie sygnałów cyfrowych, natomiast sygnały analogowe z czujników pomiarowych są przekazywane do regulatora poprzez przetwornik analogowo-cyfrowy (A/D[27]). Do zalet tej grupy układów należy możliwość realizacji dowolnie złożonych algorytmów sterowania (włącznie ze sterowaniem optymalnym i ad-

[26]ang. *Direct Digital Control.*
[27]ang. *Analog/Digital.*

aptacyjnym). Układy te pozwalają na ciągły pomiar i rejestrację wartości dowolnych parametrów procesu, przetwarzanie danych pomiarowych, wykrywanie i sygnalizację stanów awaryjnych, większa dokładność sterowania, wynika z precyzyjnej identyfikacji obiektu regulacji. DDC może pełnić tę samą funkcję jak regulator konwencjonalny, ale także potrafi zastąpić większą liczbę układów regulacji konwencjonalnej.

Zadaniem systemów HVAC jest zarządzanie komfortem osób przebywających w pomieszczeniach objętych działaniem systemu. Komfort związany jest np. z zapewnieniem odpowiedniej jakości powietrza poprzez odczytywanie czujników ciągłego monitorowania temperatury i wilgotności oraz wysterowywanie układów wykonawczych (służących zapewnieniu prawidłowej wymiany powietrza). Układy wykonawcze automatyki bezpośrednio sterują powietrzem poprzez wydmuchanie jego strumienia na zewnątrz budynku, ogrzewanie, chłodzenie, osuszanie lub nawilżanie. W ten sposób też przeprowadza się automatyczne oczyszczanie powietrza, dodanie do niego innych składników, mieszanie oraz doprowadzanie do wybranych miejsc. Sterownie powietrzem jest istotne nie tylko z powodów zdrowotnych (m.in. zapobiega rozwojowi bakterii, wirusów, pleśni itd.), ale także pozytywnie wpływa na utrzymanie w dobrej kondycji samego budynku. Z doświadczenia widomo, że właściwe środowisko wewnątrz budynku wpływa na większość wydajność pracowników, koncentrację studentów, a nawet przyspiesza proces rekonwalescencji u pacjentów. Optymalne sterowanie urządzeniami HVAC wiąże się z integracją systemu z czujnikami systemu IB. Nie wystarczają tu jednie czujniki temperatury i wilgotności, proces optymalizacji kosztowej wymaga bowiem wzięcia pod uwagę takich czynników, jak np. pora dnia oraz roku, kąt padania promieni słonecznych na budynek czy siła wiatru.

Poprawnie skonfigurowany system HVAC powinien reagować na obecność lub nieobecność ludzi w pomieszczeniach. W pomieszczeniach, które nie są używane, nie ma konieczności zachowania wszystkich parametrów jakości powietrza i wysokiego komfortu termicznego, czy wilgotności. Jeśli przez ustalony czas nie będzie nikogo w pomieszczeniu, wówczas czujnik obecności (zintegrowany w ramach IB/BMS) pozwoli na zredukowanie temperatury do poziomu oszczędzanego przy zachowaniu minimalnych wymagań jakościowych i termicznych. Pojawienie się osób w pomieszczeniu spowoduje natychmiastowe wejście układu sterowania w stan aktywny i podniesienie temperatury do poziomu komfortowego. Przy nieobecności użytkowników również klimatyzacja i wentylacja nie muszą pracować z wysoką intensywnością i wydajnością, aby nie marnować energii (agregaty i wentylatory są ustawiane na minimalną moc).

Na uwagę zasługuje również sposób zmiany trybu pracy systemu HVAC, który nie wymaga obecności w budynku. Również dostęp zdalny pozwala na korektę nastaw sterownika w celu optymalizacji jego pracy. Jest on szczególnie przydatny w kontekście optymalizacji parametrów środowiskowych i współdzielenia wszystkich urządzeń, a także łatwiejszego zarządzania wieloma systemami w różnych lokalizacjach jednocześnie przez jedną osobę [1].

Rozwój systemów HVAC został silnie zdeterminowany oszczędnościami wynikającymi z optymalnego użytkowania tej technologii. Systemy cieplne i wentylacyjne zużywają największe ilości energii, dlatego odpowiednia konfiguracja przynosi duże oszczędności, nie tylko finansowe, ale również klimatyczne. Na podstawie badań przeprowadzonych przez Międzynarodową Agencję Energetyczną (IEA[28]) poziom zużycia energii elektrycznej w sektorze budowlanym zwiększył się ponad dwukrotnie z poziomu mierzonego w roku 1971 do poziomu zaobserwowanego w 2010 roku. Odpowiedzią na rosnące zużycie energii w tych sektorze jest trend budowy domów zeroenergetycznych (ZEB[29]). Jest to rozwiązanie polegające na tworzeniu budynków o zerowym zużyciu energii netto i zerowej emisji dwutlenku węgla rocznie. W takich realizacjach energia wytwarzana jest lokalnie, dzięki wykorzystaniu energii pochodzącej ze źródeł alternatywnych, takich jak słońce i wiatr, przy jednoczesnym zmniejszeniu całkowitego zużycia energii z wysoko energooszczędnymi systemami ogrzewania, wentylacji, klimatyzacji i technologii oświetleniowych [80, 42, 4]. Takie podejście staje się coraz bardziej praktyczne i często obecne w nowych realizacjach. Jest też ono szczególnie uzasadnione spadkiem cen alternatywnych rozwiązań energetycznych, oraz wzrostem cen tradycyjnych paliw kopalnych. Problematyczne w przypadku implementacji aktualnych wersji systemów HVAC w obiektach typu ZEB są ich ograniczone możliwości rozbudowy oraz adaptacji do konkretnych zadań związanych z wykorzystaniem budynku. Ciekawym rozwiązaniem tego problemu jest wykorzystanie urządzeń pomiarowych działających w standardzie IoT[30] do ciągłej rekonfiguracji systemu (w zależności od pojawiających się potrzeb). W przypadku I-HVACS[31] istnieje możliwość definiowania wydzielonych stref objętych działaniem takiego system. Dane pochodzące z czujników trafiają do wspólnej bazy, gdzie poddawane są ciągłej analizie, w wyniku której generowane jest odpowiednie sprzężenie zwrotne lub sygnał sterujący [72, 51, 65, 64].

Obserwujemy dynamiczny rozwój technologii związanej ze sterowaniem w systemach HVAC. W pracach badawczych uwzględnia się urządzenia pomiarowe pracujące w standardzie IoT [96], opartym na ustandaryzowanej komunikacji pomiędzy urządzeniami a siecią Internet. Wspólna sieć pozwala na gromadzenie, przetwarzanie lub wymianę danymi. Dane pomiarowe w takich rozwiązaniach przechowywane są w jednym centralnym miejscu, gdzie łatwo jest poddać je analizie (np. w celu korekty nastaw sterowania adaptacyjnego HVAC).

Przykładem podobnej realizacji jest zdecentralizowany system sterowania wyposażony w grupy inteligentnych sensów opisany w [96]. Autorzy wykorzystują platformę

[28] ang. *International Energy Agency.*
[29] ang. *Zero Energy Building.*
[30] ang. *Internet of Things.*
[31] I-HVACS - [72]

IoT do budowy narzędzia dokonującego obliczeń w chmurze (*cloud processing*[32]). System działa w oparciu o algorytm RNN[33], estymując liczbę osób znajdującą się w danym pokoju. Model zrealizowany z wykorzystaniem RNN analizuje koncentrację CO2, temperaturę powietrza trafiającego do systemu HVAC oraz temperaturę panującą w danym pomieszczeniu. Na wyjściu modelu generowana jest informacja dotycząca liczby osób znajdujących się w badanym pokoju. Analiza tej informacji pozwala na podjęcie decyzji o zmianie nastaw układu sterowania. Z przeprowadzonych badań [96] wynika, że zysk w zużyciu energii jest na poziomie 27.12% w porównaniu do typowego systemu regułowego.

Kolejne rozwiązanie wspomagające działanie systemu sterowania HVAC, poprzez wykorzystanie urządzeń działających w oparciu o protokół IoT, pojawia się w [96], gdzie autorzy prezentują grupy komunikujących się niskoprądowych urządzeń bezprzewodowych. Pracują one w standardach takich jak WiFi[34], BLE[35] czy ZigBee[36]. Informacje pochodzące z czujników trafiają do urządzenia agregującego (*sensor box*). Ciekawym podejściem jest dostosowywanie (przez zadawanie warunków pracy) urządzeń HVAC do (rozpoznanych) konkretnie osób przebywających w pomieszczeniu. System odczytuje np. unikalny identyfikator przypisany do telefonu komórkowego. Na podstawie rozpoznanego ID, urządzenie nadzorujące zmienia nastawy systemu sterowania HVAC według zadeklarowanych parametrów.

Platforma SMO[37] jest kolejną grupą rozproszonych systemów pozwalającą na ciągłą kontrolę parametrów fizycznych jak np. (temperatura czy wilgotność) występujących w inteligentnych obiektach. System może komunikować się za pomocą standardu ZigBee pozwalając na realizację interakcji (sprzężenia zwrotnego) poprzez odseparowane galwanicznie obwody wykonawczych. Opracowane rozwiązanie jest konfigurowalne pozwalając na przełączanie go w różne tryby pracy. Szczegółowy opis platformy SMO przedstawiony zostanie w dalszej części pracy [59].

2.2.9. Protokoły komunikacyjne w IB

Protokoły komunikacyjnie otworzyły zupełnie nowe możliwości, nie tylko w automatyce, ale też w innych branżach. Metody zbierania i przesyłania danych z urządzeń funkcjonujących w ramach różnych instalacji budynkowych wymusiły potrzebę integracji, co z kolei stało się podstawowym wymogiem stawianym obecnie systemom automatyki budynkowej [50].

[32]Cloud processing – przetwarzanie w chmurze [40].
[33]ang. *Random Neural Network*.
[34]ang. *Wireless Fidelity*.
[35]ang. *Bluetooth Low Energy*.
[36]ZigBee - https://en.wikipedia.org/wiki/Zigbee.
[37]SMO - Sieciowy Monitor Obiektu.

Ewolucja systemów BAS[38] w dużym stopniu wywołana została podwyżkami cen energii w latach 70 i 80. Wzrosło wówczas też zapotrzebowanie na efektywne energetycznie budownictwo, które również napędzało rozwój systemów automatyki budynkowej. W początkowej fazie większość systemów była niezależna od siebie i dedykowana do określonych zastosowań. Brakowało narzędzi pozwalających na wzajemną komunikację, agregację danych w centralnym miejscu oraz integrację mechanizmów odpowiedzialnych za sterowanie. Brak jednolitych standardów ograniczał też możliwości komunikacji pomiędzy urządzeniami pochodzącymi od różnych producentów i dostawców. Przełomem okazały się prace nad otwartymi protokołami, które umożliwiałby pełna integrację systemów. W 1987 r. powstała organizacja BACnet[39] Project Standards Commettee, której prace badawcze realizowane przy stowarzyszeniu ANSHRAE[40] przyczyniły się do opracowania otwartego protokołu pozwalającego na integrację urządzeń pracujących w różnych systemach, wytwarzanych przez niezależnych producentów. Opracowane rozwiązanie zostało zatwierdzone i wprowadzone do powszechnego użytku przez amerykański instytut standaryzacji ANSI[41].

Standard BACnet reprezentuje protokół komunikacyjny, w którym określone są zasady rządzące wymianą danych pomiędzy urządzeniami (komunikaty i rozkazy dla urządzeń) oraz określa wykorzystywany rodzaj medium sieciowego. Za szczególną przydatnością tego rozwiązania (w kontekście wykorzystania go do sterowania budynkami) przemawia, że jego reguły i zasady komunikacji zawierają standardowe, specjalizowane zapytania i rozkazy potrzebne w tego typu instalacjach (np. konstrukcja zapytania o poziom temperatury, definicja reżimu i sekwencji czasu pracy wentylatorów, wysłanie alarmu o nieprawidłowym stanie pracy pompy itp.). Kontrolowany rozwój protokołu wraz z kolektywną pracą różnych producentów przyczynił się do użytecznego postępu w funkcjonalnej integracji systemowej pomiędzy urządzeniami. Dodatkowym motorem napędowym stała się niezależna instytucja BTL[42], która ocenia i testuje produkty w aspekcie poprawności wykorzystanie standardu. Zatem produkty oznaczony logiem BTL oznacza, że został wszechstronnie przetestowany w oparciu o obowiązujące standardy ANSI/ANSHRAE.

Możliwe jest wykorzystanie pięciu technologii w tworzeniu sieci automatyki (dla zarządzania budynkami i obiektami) opartej na protokole BACnet:

- Ethernet 10/100 Mbps – jeden z najbardziej popularnych standardów do wymiany informacji w formacie cyfrowym, która umożliwia budowę sieci komputerowych. BACnet wykorzystuje sieć CAT5[43], Ethernet lub protokoły bezprzewodowe. W

[38] ang. *Building Automation System.*
[39] ang. *Building Automation and Control Network.*
[40] ang. *American Society of Heating, Refrigerating and Air-Conditioning Engineers.*
[41] ang. *American National Standards Institute [43].*
[42] ang. *BACnet Testing Laboratories.*
[43] https://en.wikipedia.org/wiki/Category_5_cable.

rozwiązaniach przemysłowych sieci BACnet IP zwykle oddzielone są od głównego szkieletu sieci informatycznej. Pełna separacja pomiędzy siecią informatyczną a siecią służącą do zarządzania automatyką budynkową wpływa dodatkowo na bezpieczeństwo przesyłanych danych, nie obciążając i spowalniając jednocześnie infrastruktury przeznaczonej dla użytkowników lokalnych.

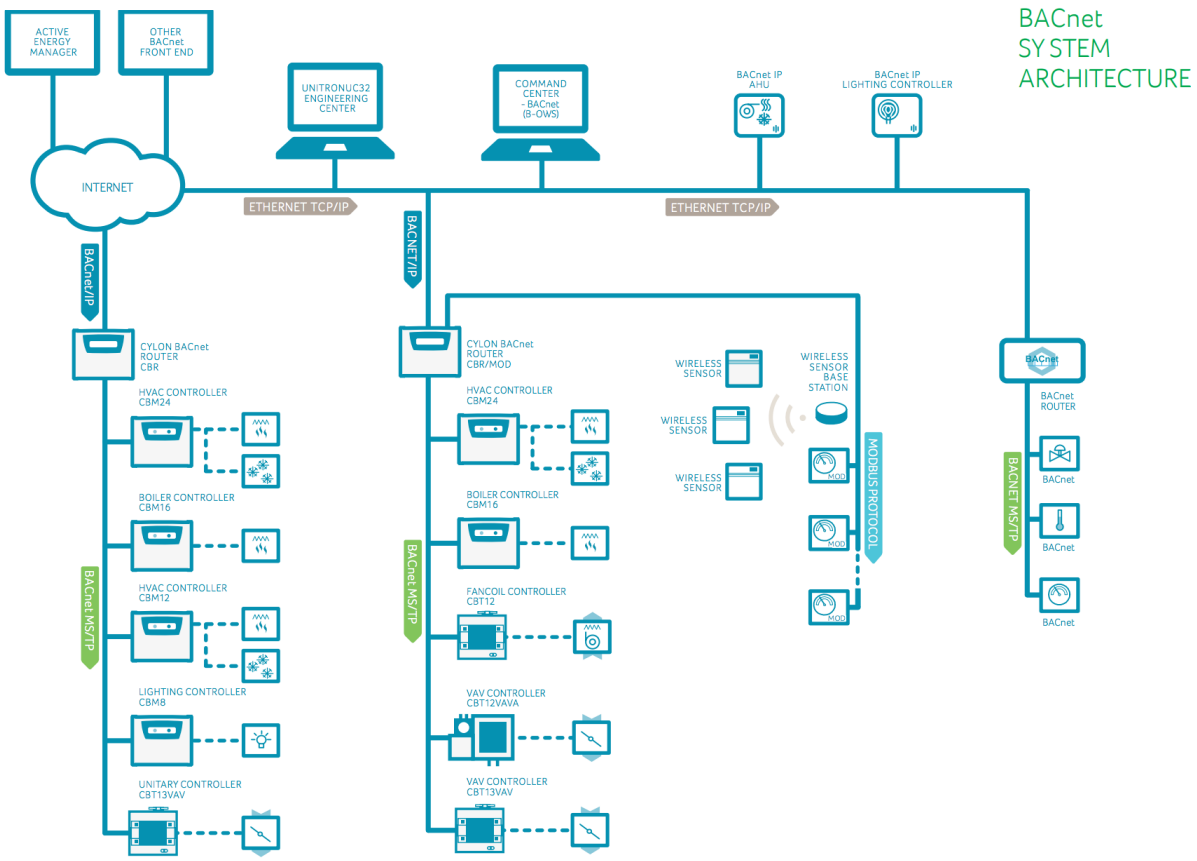
- ARCNET 2,5 Mbps – standard ANSI.
- MS/TP (master-slave/token passing) – standard ANSI, technologia transmisji w standardzie BACnet opracowana dla urządzeń o niższych wymaganiach co do szybkości transmisji (1 Mbps lub mniej). Często medium transmisyjnym jest ekranowana skrętka z buforem EIA-485, stanowiąca warstwę fizyczną.
- LonTalk 1,25 Mbps – protokół transmisyjny opracowany przez firmę Echelon (oryginalnie dedykowany dla sieci LonWorks).
- PTP (point to point) szybkość do 56 kbps – protokół zdefiniowany specjalnie dla systemu BACnet, umożliwia komunikację z wykorzystaniem tradycyjnej linii telefonicznej [24, 76, 93].

Opisane wyżej standardy definiują sposoby komunikacji w sieciach automatyki budynkowej. Podsystemy łączą się w sieć za pomocą routerów[44], tworząc jedną infrastrukturę. W przypadku, kiedy systemy pracują w różnych standardach telekomunikacyjnych w celu ich wzajemnej integracji, stosuje się specjalistyczne bramy łączeniowe (gateways). Wiąże się to z inżynierią topologii sieci i mechanizmów adresowania oraz wykorzystaniem protokołów komunikacyjnych [66]. Sposoby komunikacji pomiędzy urządzeniami zilustrowane zostały na rys. 2.6.

BACnet jest obecnie jednym z najczęściej wykorzystywanych protokołów w Stanach Zjednoczonych, gdzie zdecydowana większość producentów urządzeń automatyki budynkowej, wytwarza je zgodnie z tym standardem [4, 82].

Największym konkurentem dla BACnetu jest standard **LonWorks** opracowany przez fimę Echelon Corp. Został on opracowany na przełomie lat 80. i 90. XX wieku w Stanach Zjednoczonych. Echelon został powołany do istnienia przez światowe przedsiębiorstwa Motorola i Toshiba, w celu stworzenia otwartej technologii umożliwiającej komunikację w rozproszonych systemach automatyki [95]. Podobnie jak w przypadku BACnetu twórcom tego rozwiązania też zależało na opracowaniu standardu komunikacji pomiędzy poszczególnymi urządzeniami systemu sterowania budynku, pochodzącymi od różnych producentów. Kompatybilność oraz standaryzację urządzeń współpracujących w oparciu LonWorks zapewnia organizacja LonMark, która zrzesza wytwórców oraz instalatorów urządzeń tworzonych w tej technologii [92, 68]. Opracowane przez nich system

[44]Router - https://en.wikipedia.org/wiki/Router_(computing).

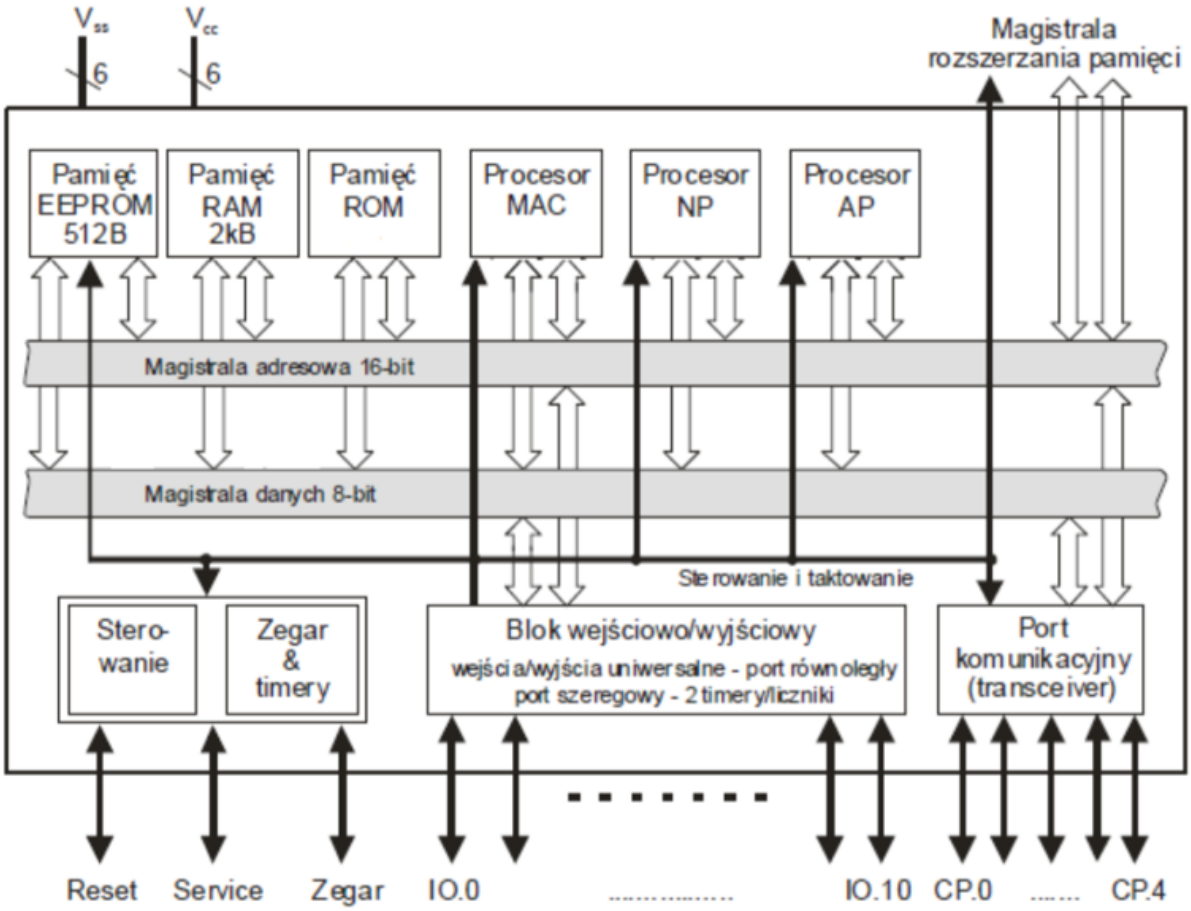


Rys. 2.6. Schemat sieci opartej o protokół BACnet [48].

standaryzacji pozwala na pełna integrację urządzeń tworzących sieć urządzeń do automatycznego monitorowania stanu budynku oraz sterowania w ramach jednego systemu elektronicznego i teleinformatycznego. Sieć automatyki budynkowej (LON[45]) pod wieloma względami (funkcji, parametrów) przypomina znane i powszechnie wykorzystywane sieci LAN[46]. Proces komunikacji pomiędzy urządzeniami odbywa się z wykorzystaniem protokołu LonTalk w trybie pakietowym P2P (*peer-to-peer*). Podstawową jednostką w sieci są urządzenia zwane węzłami (ang. *node*), które w sieci mogą być reprezentowane przez czujnik, kontroler a nawet urządzenie służące do zaawansowanego przetwarzania danych. Elementem składowym każdego węzła jest mikroprocesor NeuronChip o architekturze przedstawionej na rys. 2.7.

Mikroprocesor Neuron zawiera w swojej strukturze trzy ośmiobitowe jednostki CPU, z których każda ma inne funkcje:

[45]ang. *Local Operation Network*.
[46]ang. *Local Area Network*.



Rys. 2.7. Architektura mikroprocesora NeuronChip [67].

- Procesor MAC[47] – realizuje algorytm dostępu do mediów transmisyjnych oraz jest odpowiedzialny za komunikację w sieci, jak również za obsługę podsystemu do wykrywania i unikania kolizji.
- Procesor sieciowy Network CPU – obsługuje zmienne sieciowe, liczniki, adresuje przesyłki, obsługuje podsystemu diagnostyki i zarządzania w sieci. Wymiana danych z procesorem MAC odbywa się z wykorzystaniem buforu sieciowego, natomiast z procesorem AP poprzez bufor aplikacji.
- Procesor aplikacji AP (Application Procesor) – wykonuje programy napisane w języku NeuronC [92, 82].

Kolejnym standardem istotnym dla inteligentnego budownictwa (aplikacji sprzętowych wykorzystujących sterowniki i czujniki) jest protokół **Modbus**. Do komunika-

[47] ang. *Media Access Control Address.*

cji pomiędzy urządzeniami można wykorzystywać RTU[48], z zastosowaniem ekranowanej skrętki oraz buforu EIA-485, jak również struktury TCP/IP (Etherent, CAT5E itp.) z przeznaczeniem do pracy zarówno w sieci intranet, jak i w Internecie. Wśród typowych zastosowań tego standardu są rozwiązania przemysłowe takie jak np. pomiary energetyczne, sterowanie napędami czy też klasyczne systemy sterowania. Integracja z większością innych standardów do komunikacji nie stanowi większego problemu dzięki zastosowaniu standardowych bram sieciowych (*gateways*) [4].

EIB/KNX to kolejny popularny standard koordynujący pracę urządzeń działających w inteligentnych budynkach, opracowany już w latach 90. XX wieku przez konsorcjum złożone z firm działających w branży instalacji elektrycznych i automatyki - stowarzyszenie EIBA[49], które zrzesza obecnie około 100 firm zajmujących się budową produktów działających zgodnie ze standardem EIB/KNX. Duża liczba współpracujących ze sobą podmiotów sprawia, że EIB/KNX jest najpopularniejszym systemem automatyki budynkowej w Europie. Etymologia zmiany nazwy standardu z początkowego EIB związana jest z przekształceniem strukturalnym w stowarzyszeniu mającym miejsce na przełomie 2003/2004 roku. Stowarzyszenie zmieniło wówczas nazwę na KONNEX, modyfikując jednocześnie nazwę standardu na EIB/KNX. Geneza powstania systemu była bardzo zbliżona do opisanego wcześniej standardu LonWorks – chodziło mianowicie o stworzenie zdecentralizowanego systemu do załączanie sterowania, regulacji i nadzóru urządzeń elektrycznych i elektrotechnicznych stanowiące wyposażenie budynku oraz jego otoczenia. Koncepcja ta jest bardzo zbliżona do opisywanych wcześniej poprzedników. Magistrala stanowi medium do komunikacji, natomiast każde urządzenie posiada wbudowany układ pozwalający na wymianę informacji pomiędzy nim a magistralą. Zaletą tego standardu jest stosunkowo duża różnorodność urządzeń pracujących w tym standardzie, oferowanym przez wielu dostawców.

Magistrala stanowi zatem podstawowe medium umożliwiające komunikację pomiędzy urządzeniami magistralnymi (*bus devices*), które mogą pełnić w sieci funkcje czujnika, aktora albo elementu sterującego lub logicznego (realizacja funkcji logicznych). Urządzenia magistralowe zbudowane są z trzech podstawowych elementów: moduł łączeniowy BCU[50], moduł aplikacji AM[51] oraz program aplikacyjny AP[52]. Schemat urządzenia magistralnego przedstawiono na rys. 2.8.

Moduł BCU pozwala na połączenie urządzenia magistralowego z modułem aplikacji poprzez zastosowanie interfejsu PEI[53] (najczęściej złącze 10-cio pinowe). Moduł BCU może stanowić integralną część urządzenia magistralowego lub występować oddzielnie

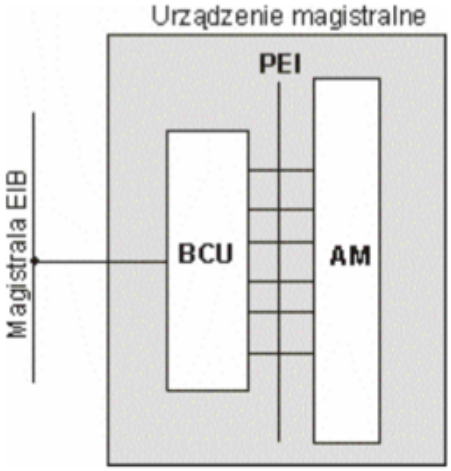
[48]ang. *Remote Terminal Unit.*
[49]ang. *European Installation Bus Association.*
[50]ang. *Bus Coupling Unit.*
[51]ang. *Application Module.*
[52]ang. *Application Program.*
[53]ang. *Physical External Interface.*

(nazywany wtedy BAU[54]). Głównym zadaniem realizowanym przez moduł BCU jest dekodowanie przesłanych magistralą telegramów na odpowiednie sygnały sterujące modułem aplikacji w sytuacji, kiedy urządzenie magistralne pełni rolę aktora (członu wykonawczego). W trybie pracy czujnika, urządzenie magistralne koduje i generuje odpowiednie telegramy do sieci sterowania na podstawie sygnałów odbieranych z modułu aplikacji. Każdy z wykorzystywanych w standardzie modułów BCU zawiera następujące elementy:

- Transceiver – pośredniczy w komunikacji pomiędzy modułem BCU a magistralą EIB/KNX.
- Kontroler BCC[55] – zbudowany z mikrokontrolera, przetworników A/C[56] oraz kilku rodzajów pamięci, wśród których wyróżniamy trzy rodzaje pamięci: pamięć ROM[57] przechowującą informacje i ustawienia producenta, pamięć EEPROM[58] umożliwiającą zapisywanie programów aplikacyjnych jak również pamięć RAM[59] zawierającą aktualne wyniki obliczeń. W skład takiego kontrolera wchodzą również układy I/O[60], interfejs szeregowy oraz element taktujący. Urządzenia KNX/EIB obsługują obecnie trzy mikrokontrolery firmy Motorola:
  - Mikrokontroler Motorola 68HC05B6 – interfejs BIM[61] M 111 lub jednostka BCU 1, układ wyposażony został w przycisk programowy oraz informacyjną diodę LED.
  - Mikrokontroler Motorola 68HC05BE12 – interfejs BIM M 113, lub jednostka BCU 2, opracowana przez Motorolę do typowych rozwiązań wykorzystujących standard KNX/EIB, posiada zwiększoną pamięć ROM, RAM, EEPROM (12 kbajtów pamięci ROM). Obsługuje wszystkie aplikacje programowe zgodne z BCU 1 oraz komunikację przez obiekty KNX/EIB.
  - Mikrokontroler Motorola 68HC11E9 - interfejs BIM M 112 lub jednostka BCU 3 (z architekturą różną od BCU 1 i BCU 2), stanowi wyspecjalizowany układ sterujący pracujący (natywnie) w standardzie KNX/EIB, który pozwala na implementację zaawansowanego sterowania i kontroli (aplikacje symulacyjne, funkcje logiczne i czasowe).

[54]ang. *Bus Access Unit.*
[55]ang. *Bus Coupling Controller.*
[56]ang. *przetwornik Analogowo-Cyfrowy.*
[57]ang. *Read Only Memory.*
[58]ang. *Electrically Erasable Programmable Read-Only Memory.*
[59]ang. *Random-Access Memory.*
[60]ang. *Input/Output.*
[61]ang. *Bus Interface Module.*

- Moduł aplikacji AM – urządzenie elektryczne, pozwalające na realizację zadań zdefiniowanych w programie aplikacyjnym AP, który może być umieszczony w pamięci ROM modułu BCU lub w pamięci modułu aplikacyjnego.



Rys. 2.8. Schemat budowy urządzenia magistralnego.

Technologie umożliwiające budowanie programów aplikacyjnych na urządzenia KNX/EIB są praktycznie nieograniczone (możliwie jest tworzenie aplikacji w wielu językach programowania). Powstałe ograniczenia i wymagania wynikają jedynie z zastosowanej jednostki centralnej (procesor) w module łączeniowym BCU lub oferowanych przez producenta narzędzi programowych (asembler, kompilator, emulator). Stowarzyszenie KONNEX oferuje producentom urządzeń zintegrowane środowisko bazujące na języku ANSI C do realizacji projektów wykorzystujących standard EIB/KNX [82, 87, 94, 88].

LoRaWAN[62] to kolejny ze standard pozwalający na budowę sieci sensorowych w inteligentnych budynkach. Specyfika jego leży w medium transmisyjnym wykorzystywanego do przesyłania danych, które jest standardem bezprzewodowy należący do rodziny LPWAN[63], opracowany z myślą o rozwiązaniach IoT. Celem opracowania koncepcji LoRA(WAN) była minimalizacja problemów związanych z ograniczonym zasięgiem oraz czasem pracy urządzeń z wykorzystaniem zasilania bateryjnego, w takich standardach jak WiFi, Bluetooth[64] czy też ZigBee. W opracowaniu uwzględniono też eliminację kosztów generowanych przez moduły GSM[65]. Główna misją technologii LoRa jest zatem realizacja komunikacji na bardzo duże odległości przy niskim koszcie zużycia energii, gdzie

[62]ang. *Long Range Wireless Network.*
[63]ang. *Low Power Wide Area Network.*
[64]https://en.wikipedia.org/wiki/Bluetooth
[65]ang. *Global System for Mobile Communications.*

dane przesyłane są w małych blokach z wykorzystaniem algorytmu szyfrującego AES[66] z kluczem 128-bitowym. Sieć działa w oparciu o protokół asynchroniczny z modulacją pozwalającą na odbiór sygnałów 22dB (poniżej progu szumów) przy zastosowaniu technologii rozpraszania widma CSS[67]. Komunikacja realizowana jest tu dwukierunkowo w trybie półdupleksu, w topologii gwiazdy.

Moduł tego standardu pozwala na adaptacyjne dopasowania mocy nadajnika i szybkości transmisji do aktualnych warunków propagacyjnych. Bilans mocowy dla urządzenia SX1272 firmy Semtech kształtuje się w sposób następujący: urządzenie pobiera przy nadawaniu od 18 mA (przy 7 dBm) do 125 mA (przy 20 dBm), 10 mA podczas odbierania i zaledwie 1,5 µA w stanie bezczynności [90]. Dane te jednoznacznie wskazują na możliwość optymalizacji zasilania bateryjnego w aplikacjach sprzętowych realizowanych w oparciu o technologię LoRa. Co więcej przy tak małym zużyciu energii można zastąpić tradycyjne zasilanie bateryjne rozwiązaniami z klasy energy harvesting, polegającymi na wykorzystaniu energii pozyskiwanej z niewielkich ogniw fotowoltanicznych, bądź też z elementów piezoelektrycznych. Wspomniane wcześniej transceivery SX1772 mają zasięg do 5 km w środowisku miejskim oraz do 15 km w terenie otwartym [89]. Pozwala to na objęcie działaniem kilku takich modułów stosunkowo dużej powierzchni. Dodatkową zaletą rozwiązania jest możliwość pracy w nielicencjonowanym paśmie częstotliwości ISM (433 MHz, 868 MHz oraz 915 MHz). Jak wcześniej wspomniano głównym ograniczeniem standardu jest szybkość transmisji danych, która waha się pomiędzy 0,3 a 50 Kbps [69]. Jednak zasięg oraz bardzo niskie zapotrzebowanie na energię sprawią, że rozwiązanie to doskonale nadają się do budowy rozproszonych systemów pomiarowych [50].

Opisane wyżej standardy komunikacyjne odpowiadają też technologiom przetwarzania w chmurze[68] oraz integracji z urządzeniami mobilnymi (IoT). Niewątpliwą zaletą przechowywania i przetwarzania danych pomiarowych w chmurach obliczeniowych jest możliwość wykorzystania praktycznie nieograniczonych możliwości archiwizacji danych oraz skalowania infrastruktury sprzętowej, na której prowadzimy obliczenia. Nowoczesne rozwiązania programistyczne pozwalają przy tym w stosunkowo łatwy sposób zapanować nad wizualizacją i analizą danych w czasie rzeczywistym.

2.3. Diagnostyka rozproszonych obiektów przemysłowych

Olbrzymie możliwości analizy oraz agregacji danych pochodzących z systemów pomiarowych, występujących w inteligentnych budynkach, pozwalają na budowę coraz to bardziej

[66] ang. *Advanced Encryption Standard.*
[67] ang. *Chirp Spread Spectrum.*
[68] ang. *cloud computing.*

zaawansowanych systemów diagnostycznych. Stopień skomplikowania tych systemów skorelowany jest z rozwojem i złożonością technologiczną nowoczesnych budynków mieszkalnych i przemysłowych. Wykorzystanie strategii diagnostycznych w IB jest szczególnie istotne w kontekście systemów energetycznych, które silnie uzależnione są od sprzętowych i programowych komponentów tj. sensory czy aktuatory. Występowanie błędów odczytu czy też generowanie niewłaściwych reakcji systemu, prowadzić może do niewłaściwego działania podsystemów, a w konsekwencji do generowania dużych start przez główne systemy energetyczne w dużych budynkach przemysłowych. Zachodzą tu bowiem długie okresy, w których system działa niepoprawnie, wykorzystując błędne wskazania urządzeń i czujników. Oszacowano, że w wyniku pojawiających się błędów koszty związane ze zużyciem energii w budynkach ulegają zwiększają się od 15% do 30% [49]. Rozwiązanie tego problemu opiera się na zautomatyzowanym procesie wykrywania błędów i diagnostyki (FDD[69]), pozwalającym na odkrywanie potencjalnych problemów oraz szybsze i dokładniejsze przewidywanie, kiedy i co może ulec awarii.

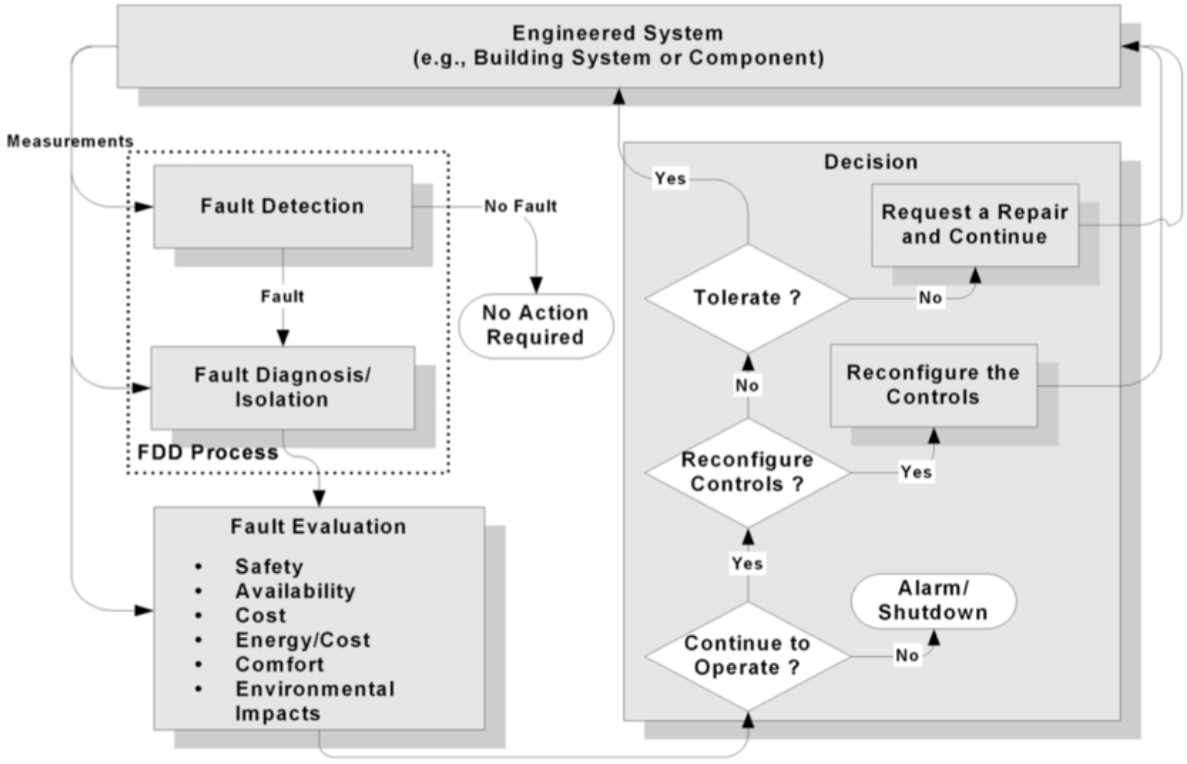
FDD daje możliwość wykrywania i zgłaszania istotnych błędów, pomagając w znalezieniu i wyizolowaniu problemów odpowiedzialnych za marnotrawienie energii, oraz we wczesnym ostrzeganiu przed zbliżającą się awarią [34].

Zautomatyzowany system FDD służący diagnostyce i sterowania inteligentnym budynkiem przedstawiono na rys. 2.9. W pierwszym kroku system dokonuje analizy danych pomiarowych oraz wykrywa potencjalne błędy. Wykrycie nieprawidłowego stanu, prowadzi do uruchomienia automatycznego procesu odpowiedzialnego za diagnozowanie błędów oraz znalezienie ich przyczyn. Kolejne kroki na przedstawionym diagramie, dotyczą ewaluacji wykrytego błędu w kontekście jego wielkości i wpływu na działanie całego systemu. Ewaluacja może dotyczyć wielu aspektów, np. zwiększonego zużycia energii, dostępności, utraty komfortu lub wpływu na inne elementu systemu. W następnym etapie system FDD podejmuje reakcję zależną od powstałego błędu oraz jego klasyfikacji. W większości przypadków samo wykrycie błędów przez system jest relatywnie proste, zdecydowanie trudniejsze okazuje się zdiagnozowanie przyczyny problemu lub określenie wpływu napotkanej usterki na zachowanie pozostałej części systemu [49].

W pracy [71] przedstawiono trzy podstawowe podejścia do budowania systemów opartych na FDD: podejście modelowe (*model-based*), bazujące na danych (*data-based model*) oraz podejście hybrydowe (wykorzystujące zarówno dane jak i model).

Podejście modelowe sprawdza się najlepiej w przypadku małych obiektów (w dużych realizacjach tworzone modele bywają bardzo skomplikowane i czasochłonne). Wybór podejścia modelowego zwykle wymaga kompromisu pomiędzy stopniem skomplikowania modelu a odwzorowaniem modelowanego systemu. Problemem, który pojawia się w przypadku modelowania jest nadmierne uproszenie, która wpływa na efektywność działania

[69]ang. *Fault Detection and Diagnostics*.



Rys. 2.9. Przykład działania systemu FDD w IB [49].

FDD.

Technika oparta na danych jest zdecydowanie łatwiejsza do implementacji, ale jej dokładność – w odniesieniu do monitorowanych obiektów – uzależniona jest od dostępności oraz jakości danych. Dostęp do danych bezpośrednio wykorzystywanych w działaniach systemów FDD może jednak być trudny lub bywa ograniczony. Ponad to zbiór danych treningowych pozwalający na uczenie modelu, powinien zawierać informacje o symptomach awarii. W podejściu tym uczymy system wczesnego wykrywania powtarzających się błędów – pozostają jednak zdarzenia, które wcześniej nie występowały w zbiorach treningowych.

Diagnostyka z wykorzystaniem modelowania danych bazuje na grupie algorytmów związanych z uczeniem maszynowym (ML[70]). Jak wspomniano wcześniej, algorytmy trenowane są danymi archiwalnymi, aby później, w czasie rzeczywistym, potrafiły dokonywać analizy danych i predykcji awarii. Podejście wykorzystujące modelowanie w systemach FDD za pomocą danych zaprezentowano w [73]. Autorzy do wyodrębnienia informacji o błędach analizują główne komponenty systemu z wykorzystaniem algorytmów

[70]ang. *Machine Learning*.

generycznych, SVM[71] oraz regresji PLS[72]. Ciekawe podejście do analizy danych pomiarowych w celu predykcji związanej z pojawiającymi się błędami stwarza oprogramowanie opisane w [23], gdzie autorzy zastosowali w analizie danych algorytmy genetyczne oraz algorytm regułowy QAR[73].

W podejściu modelowym (*model-based*) istotny jest odpowiedni sposób regulacji (czyli metoda i parametry sterowania) oraz dobór poziomu kwantyzacji w celu realizacji sterowania przez układy wykonawcze. W przypadku złożonych (hierarchicznych) systemów dokonuje się upraszczających założeń. Jednym z założeń jest przypisywanie deterministycznego charakteru działania dla modelowanego obiektu lub komponentu (w tym zasada pewnościowej równoważności – ang. *certainty equivalence*). Takie modelowe podejście przedstawiono w [81], gdzie zaprezentowano system czasu rzeczywistego do monitorowania i diagnostyki rozproszonych systemów energetycznych. W publikacji tej odpowiednie modelowanie przyczyniło się do uzyskania na poziomie 30% oszczędności w konsumpcji energii.

W podejściu hybrydowym korzystamy z obu metod w opracowywaniu rozwiązania końcowego. Przykładowe rozwiązanie przedstawiono w [19]], gdzie autorzy wykorzystują falkowe sieci neuronowe (*wavelet neural network*), będące połączeniem sieci neuronowych i analizy falkowej. System został zbudowany do analizy danych pochodzących z sensorów (temperatura, średni przepływ, ciśnienie) w systemie VAV[74], stanowiącym część systemu HVAC. Przedstawiono tam również porównanie metody hybrydowej z poszczególnymi składowymi (model oraz data-driven), z którego wynika, że największą skuteczność działania systemu diagnostycznego otrzymujemy w podejściu hybrydowym.

Największym problemem występującym przy wszystkich podejściach jest brak umiejętności wczesnego wykrywanie błędów, których rozróżniania system nie został nauczony. Praktycznie niemożliwe jest przygotowanie zbioru treningowego, zawierającego wszystkie możliwe sytuacje, z błędami. Szczególnie, że rozważamy duże systemy przemysłowe, często rozproszone po wielu obiektach jednocześnie. Jest to temat ciągle aktywny naukowo, gdyż nadal prowadzone są badania dotyczące identyfikacji błędów niewystępujących w zbiorach uczących. Metodę pozwalającą na identyfikację takich błędów z wykorzystaniem sieci neuronowych oraz metod grupowania przedstawiono w [18].

Uczące systemy FDD posiłkują się również informacją pochodzącą bezpośrednio od użytkowników. W przypadku inteligentnych budynków informacja trafiająca do systemu diagnostycznego może być wzbogacana wiedzą od administratorów, jak również inżynierów procesu zarządzających automatyką obiektów. Sukcesywnie dostarczana wiedza, dobrze wpływa na systemy sterowania w IB. Bezpośrednim wynikiem takiego podejścia

[71] ang. *Support Vector Machine.*
[72] ang. *Partial Least Squares.*
[73] ang. *Quantitative Association Rules.*
[74] ang. *Variable Air Volume.*

FDD jest pojawienie się bezpośredniej korelacji pomiędzy wydajnością działania samego budynku a komfortem korzystających z niego użytkowników [71].

Nowatorskie podejście budowania systemów FDD w inteligentnych budynkach przedstawiono w [71], gdzie autorzy proponują koncepcję centralnego systemu opartego o chmurę obliczeniową. Komunikacja z obiektami (poprzez sensory i aktuatory) realizowana jest wówczas w architekturze serwisowej (SOA[75]) [5], która polega na korzystaniu z wielu uniwersalnych komponentów udostępniających określone funkcjonalności, najczęściej w środowiskach rozproszonych.

Duża zaletą rozwiązań chmurowych jest ich skalowalność – w przypadku kończących się zasobów sprzętowych istnieje możliwość łatwego rozszerzenia parametrów środowiska serwerowego w którym uruchamiana jest aplikacja. Skalowalność sprawia również, że zdecydowanie szybciej dochodzimy do rozwiązania problemów wymagających obliczeniowo. Przekłada się to bezpośrednio na szybszą diagnostykę, jak również możliwość wygenerowania sygnału sterującego w krótszym czasie. Serwisowa architektura oprogramowania ułatwia dopasowanie zadań diagnostyczne do konkretnego budynku. W ten sposób można zbudować jeden system diagnostyczny dla wielu inteligentnych budynków jednocześnie. Niewątpliwą zaletą takiego podejścia jest również możliwość dokonywania analizy porównawczej pomiędzy systemami budynkowymi o podobnych cechach, oraz opracowywania systemów o doskonalszej filtracji danych pomiarowych, predykcji, oraz wyższej wydajności.

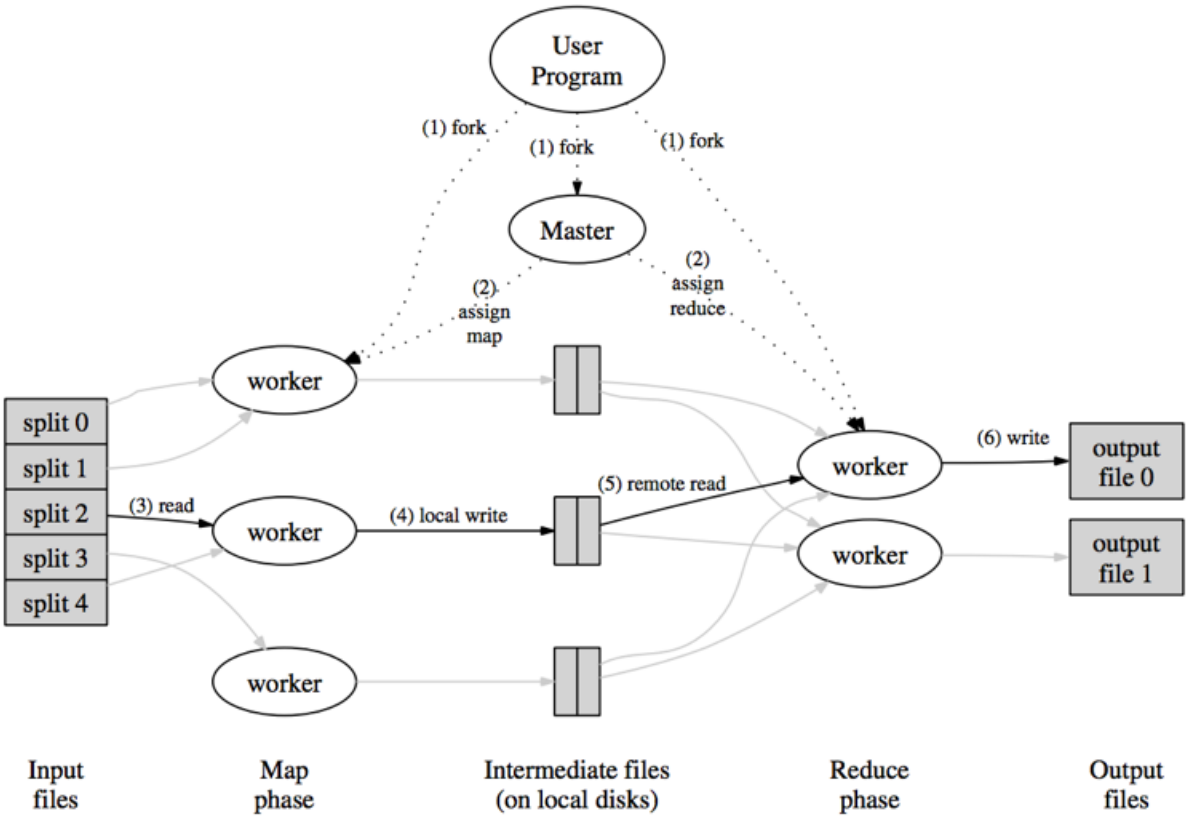
2.4. Systemy bazodanowe w nadzorowaniu inteligentnych budynków

Organizacja ICD szacuje, że liczba otaczających nas cyfrowych danych w 2013 roku liczyła 4,4 zettabajta[76], a do roku 2020 wielkość ta ma wzrosnąć dziesięciokrotnie [99]. Składową tej wielkości są również bardzo duże ilości generowanych informacji przez systemy pomiarowe i diagnostyczne. Jednym z przykładów są dane pomiarowe pochodzące ze Zdrzacza Hadronów zlokalizowanego pod Genewą w Szwajcarii, który generuje około 30 petabajtów danych rocznie. Taka masa informacji wymaga wyspecjalizowanych narzędzi pozwalających na ich odpowiedni, ustandaryzowany zapis oraz efektywny odczyt. Wszystko to wpłynęło na potrzebę opracowania zupełnie nowych metod przechowywania danych w zoptymalizowanych i ustandaryzowanych strukturach.

Relacyjne systemy baz danych przestały być jedyną wykorzystywaną formą zbierania informacji – nie tylko ze względu na swoją restrykcyjną strukturę, ale przede wszystkim ze względu na ograniczone możliwości skalowania. Ponieważ w dużych systemach liczba

[75]ang. *Service Oriented Architecture*.
[76]zettabajt - 10^{21} bajtów, tysiąc eksabajtów, milion petabajtów, miliard terabajtów.

generowanych i analizowanych danych szybko zaczęła przekraczać możliwości składowania ich na pojedynczych komputerach, szybko rozpoczęto badania nad opracowaniem technologii umożliwiającej łączenie komputerów w klastry współdzielące pomiędzy sobą przestrzeń dyskową, pamięć operacyjną oraz CPU[77]. Taka metoda analizy i agregowania danych ciągle jest rozwijana. Powstają coraz to lepsze rozwiązania pozwalające na bardziej efektywny sposób zapisu i odczytu danych zlokalizowanych na klastrze.



Rys. 2.10. Przykład działania modelu MapReduce [13].

Klastrowe rozwiązanie wymaga również specyficznego podejścia do budowy oprogramowania uruchamianego w takim środowisku. Model MapReduce, opisany przez Deana oraz Ghemawata w artykule [13], model polega na podziale przetwarzania na dwa etapy: mapowania oraz redukcji. W każdym etapie pobiera się danej wejściowe i generuje dane wyjściowe w formie pary *klucz-wartość*. Typ zmiennej klucza i wartości określany jest przez programistę. Oczywiście programista odpowiedzialny jest również za implementację algorytmu mapowania i redukcji. Zaletą tego podejścia jest możliwość rozproszenia jednego programu na kilka wiele wątków działających na różnych komputerach jednocześnie. W analizie masowych danych, takie rozwiązanie pozwala na przyspieszenie obliczeń

[77] ang. *Central Processing Unit*.

w oparciu o zasoby sprzętowe dostępne w klastrze. Zasada ta polega na rozmieszczeniu zadania przez funkcję mapującą na wielu komputerach jednocześnie, przy czym mechanizm rozmieszczania działa automatycznie wykorzystując proces partycjonowania danych wejściowych. W jego wyniku powstaje M niezależnych grup zadaniowych, z których każda może być przetwarzana na osobnej maszynie. Krok redukcji polega na podziale przestrzeni danych zawierającej klucze pośrednie na R fragmentów, z wykorzystaniem odpowiedniej funkcji partycjonującej (np. hash(key) **mod** R). Redukcja przebiega w trybie rozproszonym, gdzie liczba partycji (R) oraz funkcja partycjonująca definiowana jest przez programistę. Sposób działania algorytmu przedstawiono na rys. 2.10 [99, 13].

Implementacje modelu MapReduce są powszechnie wykorzystywane w platformie Hadoop[78], narzędziu pozwalającym na realizację obliczeń w trybie rozproszonym. Dodatkową cechą tego rozwiązania jest możliwość działania na rozproszonym systemie plików HDFS[79], gdyż zostało ono opracowane w celu przechowywania dużych zbiorów danych, nie mieszczących się na jednej maszynie. HDFS zaprojektowano z myślą zapewnienia maksymalnego bezpieczeństwa danych poprzez zastosowanie mechanizmu replikacji, działającego na poziomie pojedynczych bloków danych. Tak jak w przypadku innych systemów plików (FAT[80], NTFS[81], EXT3[82]) również w systemie HDFS występują bloki, ale są one znacznie większe. Domyślnie mają 128 megabajtów. Duże pliki dzielone są na bloki i mogą być przechowywana na dowolnych dyskach klastra. Dodatkową zaletą tego podejścia jest możliwość replikacji, zapewniająca odporność na błędy oraz wysoką dostępność. W celu ochrony przez uszkodzeniem bloku oraz awariami dysków i maszyn, każdy blok jest replikowany na kilku (zwykle trzech) fizycznie niezależnych maszynach. Gdy blok staje się niedostępny dla klienta (uszkodzenie lub awaria komputera), można go wczytać z innej lokalizacji w klastrze w sposób całkowicie dla użytkownika [99].

Model MapReduce (MR) nie jest jedynym pozwalającym na dokonywanie obliczeń w trybie rozproszonym. Mocnym konkurentem jest środowisko Spark[83] - pozwalające w odróżnieniu do modelu MR - na przechowywanie w pamięci dużych roboczych (pośrednich) zbiorów danych. Dzięki temu możliwe jest uzyskanie wyższej wydajności niż w modelu MR (różnica w czasie wykonywania zadań MR jest o rząd większa od tożsamych zrealizowanych w modelu Spark) [99, 100, 101] Spark (podobnie jak model MR) stosuje zadania (ang. *jobs*), są one jednak bardziej ogólne niż te w modelu MR, gdyż mogą składać się z dowolnego acyklicznego grafu skierowanego etapów. W środowisku uruchomieniowym Sparka rozbija etapy na operacje (ang. *tasks*) w celu wykonywania równoległych opera-

[78]Hadoop - http://hadoop.apache.org/
[79]ang. *Hadoop Distributed File System.*
[80]ang. *File Allocation Table.*
[81]ang. *New Technology File System.*
[82]ang. *Third Extended Filesystem.*
[83]Spark - https://spark.apache.org/

cji na partycjach zbiorów RDD[84] dostępnych w klastrze. Zbiory RDD to najważniejsze abstrakcyjne obiekty Sparka przeznaczone tylko do odczyt, które w aplikacjach reprezentowane są jako kolekcje obiektów dzielone pomiędzy maszynami klastra [99].

Kolejną grupą rozwiązań wykorzystywaną do przechowywania i analizy danych pomiarowych, są bazy NoSQL[85]. Stanowią one też alternatywą dla obecnych od lat 70. postulatów Edgara Franka Codda, dotyczących zasad projektowania relacyjnych struktur bazodanowych. Bazy NoSQL podzielimy na cztery główne kategorie ze względu na wykorzystywany model: klucz-wartość, dokument, rodzina kolumn i graf. W bazach NoSQL podstawową formą pozwalającą na przechowywanie danych są agregacje, zupełnie inaczej niż w modelu relacyjnym, w którym dane podzielone są na krotki (wiersze). W podejściu agregacyjnym brak jest zdefiniowanej struktury obsługiwanych danych. W modelu relacyjnym krotki, stanowią niepodzielne (atomowe) przestrzenie służące do przechowywania danych. Niemożliwe jest zagnieżdżanie krotek wewnątrz innych krotek. Operacje (modyfikacje lub zwracanie) dokonywane są na poszczególnych wierszach. W podejściu agregacyjnym wykonuje się operacje na strukturach zdecydowanie bardziej złożonych niż zestawy krotek. Możliwe jest przechowywanie wewnątrz jednego rekordu innych rekordów lub list. Agregacje ułatwiają pracę programistom, którzy chętnie operują takimi strumieniami danych. W podejściu agregacyjnym przechowywanie danych w klastrach jest o wiele prostsze niż w modelu relacyjnym. Pojedyncze agregacje mogą być przechowywane w dowolnym miejscu, ponieważ nie mają ścisłego powiązania z resztą danych. W rozwiązaniach NoSQL są dwie główne ścieżki dystrybucji danych: replikacja i współdzielenie. W replikacji te same dane kopiowane są na wiele serwerów, zaś współdzielenie oznacza umieszczanie różnych danych na różnych serwerach. Replikacja i współdzielenie to techniki ortogonalne – można korzystać tylko z jednej lub z obu [85, 86].

W przypadku systemów rozproszonych warto wspomnieć o teorii CAP przedstawionej przez Erica Brewera w roku 2000, której głównym twierdzeniem jest to, że system może posiadać tylko dwie z trzech właściwości:

- Spójność (ang. ***C****onsistency*) – dotyczy sytuacji, w której każdy użytkownik uzyska dostęp do jednakowych danych, bez względu na to z którym węzłem systemu będzie się komunikował
- Dostępność (ang. ***A****vailability*) – związana jest ze stałą (ciągłą) dostępnością do wszystkich danych
- Tolerancje na partycjonowanie (ang. ***P****artition tolerance*) – system potrafi poprawnie funkcjonować pomimo uszkodzenia części węzłów oraz komunikacji pomiędzy nimi.

[84] ang. *Resilient Distributed Dataset.*
[85] ang. *Not Only Structured Query Language.*

Przykładowo w przypadku utraty łączności pomiędzy węzłami, nie można oczekiwać stałej dostępności do danych, przy jednoczesnym zachowaniu pomiędzy nimi spójności. Wybór interesującej pary cech nie oznacza całkowitej rezygnacji z ostatniej właściwości, może jednak być ona spełniona częściowo [85].

Jedną z częściej wykorzystywanych baz w systemach automatyki budynkowej jest baza dokumentowa MongoDB[86]. Rozwiązanie to cechuje się dużą skalowalnością, wydajnością oraz swobodną struktura zapisywanych danych, które gromadzone są w formie dokumentów JSON[87]. Ułatwia to pracę z danymi, bez konieczności ich ciągłego przetwarzania przy zapisie i odczycie. Ma to stosunkowo duże znaczenie w kontekście architektury realizowanych obecnie rozwiązań programowych z zakresu diagnostyki i monitorowania, gdzie format JSON stał się uniwersalnym narzędziem przechowywania danych. Kolejną zaletą tego rozwiązania jest duża liczba interfejsów, umożliwiających komunikację w dowolnej technologii programowania (Java, Python, C++ itd.).

Baza MongoDB wykorzystana została w budowie narzędzia wspomagającego proces zarządzania inteligentnym budynkiem. W [98], służącego integracji szerokiej gamy protokołów komunikacyjnych używanych przez urządzenia pomiarowe zaliczane do grupy IoT. Zawarty tam szczegółowy opis architektury sprzętowej i programowej instruuje, jak budować i testować opracowane narzędzia. Ciekawy element stanowią proponowane agregatory danych, stanowiące most komunikacyjny pomiędzy bezprzewodowymi sensorami (ZigBee, IEEE 802.15.4 itp.) a pozostała częścią systemu służącą do agregowania i monitorowania danych pomiarowych. Agregator przypisany jest do elementu typu *Core*, którego zadaniem jest ciągłe monitorowania stanu oraz kontrola agregatorów, jak również przesyłanie informacji pomiarowych (pochodzących z agregatów pomiarowych) do zdalnej usługi chmurowej GDAS[88]. Usługa GDAS zbiera oraz poddaje analizie dane pochodzące z różnych elementów typu Core. Według autorów jeden Core przypada na jeden inteligentny budynek. Takie rozwiązanie pozwala na monitorowania i zarządzanie wieloma budynkami z poziomu centralnego serwisu GDAS. Moduł GDAS wykorzystuje dokumentową bazę Fatty, opartą na MongoDB, jako główny zbiornik (do zapisywania) danych pomiarowych. Komunikacja z bazą odbywa się za pomocą mikroserwisów, a przesyłane komunikaty mają formę wiadomości JSON.

Praktyczny przykład [44], demonstrujący wykorzystanie rozwiązań NoSQL w inteligentnym budownictwie, dotyczy rozproszonego systemu SCADA[89], opracowanego z myślą o monitorowaniu i zarządzaniu siecią inteligentnych budynków. SCADA jako narzędzie klasy BMS, pracujące w trybie rozproszonym pozwala na realizację następujących zadań szczegółowych:

[86]MogoDB - https://www.mongodb.com/
[87]ang. *JavaScript Object Notation.*
[88]ang. *Generic Data Acquisition System.*
[89]ang. *Supervisory Control And Data Acquisition.*

- komunikacja ze sterownikami PLC, regulatorami oraz koncentratorami danych pomiarowych
- przetwarzanie zmiennych procesowych
- nadzór i nadrzędna sterowanie instalacjami technicznymi w budynku
- sygnalizacja stanów alarmowych instalacji oraz urządzeń automatyki budynkowej
- archiwizacja danych procesowych wraz z raportowaniem i analizą statystyczną
- graficzna i tekstowa wizualizacja stanów pracy urządzeń i instalacji
- automatyczna obsługa działań powtarzalnych i rutynowych.

Architekturę systemu [79] reprezentuje sieć komputerów zlokalizowanych w każdym z budynków, połączonych siecią ze sterownikami PLC i regulatorami przemysłowymi, które zapewniają komunikację z urządzeniami pomiarowymi i wykonawczymi. Zastosowana technologia OPC[90], pozwalającą na jednolity dostęp do danych bez względu na rodzaj wykorzystywanego protokołu komunikacyjnego, działa na zasadzie komunikacji klient-serwer, gdzie aplikacja kliencka udostępnia dane pomiarowe w postaci zmiennych, których wartości można kontrolować. Kontrola odbywa się poprzez API dostępne w różnych językach programowania. Twórcy opracowania zwracają uwagę, że wykorzystanie typowych relacyjnych baz danych może wiązać się z powstawaniem problemów wydajnościowych przy przetwarzaniu wielkiej liczby rekordów. Ponieważ zapisywane dane zawierają jedynie informacje o nazwie mierzonej wielkości, jej wartości oraz czasie wykonania operacji, nie wymaga się tworzenia złożonych tabel, będących w relacji z innymi obiektami bazodanowymi. W związku z tym można wnioskować, że najlepszym rozwiązaniem do przechowywania tego typu danych jest baza dokumentowa MongoDB [97].

W artykule [21] proponuje się wykorzystanie architektury chmurowej przy budowie systemów do kontroli i sterowania systemami IB. Podobnie jak w [98, 97] autorzy prezentują koncepcję budowy uniwersalnego narzędzia pozwalającego na sterowanie i monitorowanie sieci inteligentnych budynków. Na uwagę zasługuje wykorzystanie specjalistycznego serwera (Fog Layer) pozwalającego na osiągniecie wyższej jakości (QoS[91]) połączenia pomiędzy urządzeniami końcowymi klasy IoT a system wizualizacji i przetwarzania umieszczonym w usłudze chmurowej. Baza MongoDB została wybrana z bardzo podobnych względów jak w [98, 97]. Autorzy zaznaczają jednak, że rozwiązania NoSQL służą najlepiej do przetrzymywania w nich dużej liczby danych heterogenicznych (różne struktury, funkcje i modele danych).

[90] ang. *Object Linking and Embedding for Process Control.*
[91] ang. *Quality of Service.*

Tematyka rozwiązań nierelacyjnych wiąże się ze stosowanymi obecnie metodami analizy danych. W większości publikacji dotyczących wspomagania procesu zarządzania inteligentnymi budynkami pojawiają się opisy architektur wspierających analizę informacji. Gotowość do analizy masowych danych oznacza możliwość skalowania horyzontalnego przez dodawanie kolejnych maszyn serwerowych, które rozszerzą możliwości obliczeniowe środowiska.

2.5. Domenowe języki programowania

W odróżnieniu od języków programowania ogólnego przeznaczenia, takich jak C++, C# lub Java, języki domenowe należą do rozwiązań wyspecjalizowanych i ukierunkowanych na rozwiązywanie konkretnych zadań i problemów. Idea tworzenia wyspecjalizowanych języków programowania nie jest nowa. Rozwiązania DSL[92] wywodzą się bowiem z grupy mini-języków dostępnych na platformach UNIX[93] . Do tej grup zalicza się narzędzia programistyczne ułatwiające zaawansowaną pracę z tekstem (*troff, eqn, pic*), wspomagające pracę powłoki systemowej (*sed, awk, itd.*) lub budowę aplikacji (*make, yacc, lex*). Dla przykładu, użycie rozwiązań DSL pozwala na definiowanie języków opisu interfejsu użytkownika, procesu biznesowego, bazy danych lub modelowanego matematycznie zjawiska. W takich zastosowaniach języki opisu problemu będą używają ontologię danej dziedziny, znacznie ułatwiając ich wykorzystanie przez ekspertów z określonego obszaru [39, 14].

Wyróżnia się dwie grupy języków DSL, wewnętrzne i zewnętrzne [14]. Zewnętrzne tworzone są w oparciu o osobny parser i kompilator, natomiast wewnętrzne osadzone są w popularnych językach programowania (Java, Groovy, Scala czy Python), gdzie należy liczyć się z pewnymi ograniczeniami i wymaganiami języka głównego (ograniczenia leksykalne i funkcjonalne).

Rozwój języków domenowych bardzo silnie zakorzenił się w dyscyplinie naukowej jaką jest automatyka i robotyka. Gruntowny przegląd dostępnych obecnie języków oraz sposobów ich wykorzystania przedstawiono w [42]. Projektowanie, symulacja i programowanie różnorodnych systemów automatyki i robotyki wymaga wszechstronnej wiedzy wielodziedzinowej, pozwalającej na integrację systemów zarówno na płaszczyźnie koncepcyjnej, jak również technicznej. Takie wymagania przekładają się na skuteczność wykorzystania języków DSL, jako narzędzi dla ekspertów z danych dziedzin, które pozwalają na lepsze zrozumienie i weryfikację procesów. Pozwala to też na większą efektywność i uniwersalność procesu tworzenia systemów automatyki. Kolejną funkcją tej grupy rozwiązań jest wypełnienie luki pomiędzy modelowaniem a implementacją oprogramowania. Języki domenowe pozwalają na opis modelu w języku zrozumiałym dla ekspertów dziedzino-

[92] ang. *Domain Specific Language*.
[93] UNIX - http://opengroup.org/unix.

wych (np. z branży lotniczej, motoryzacyjnej, telekomunikacyjnej, czy też związanej z inteligentnym budownictwem) oraz przyspieszenie procesu implementacyjnego poprzez generowanie fragmentów kodu potrzebnego do rozpoczęcia procesu symulacji. Taka uniwersalność pozwala nie tylko na przyspieszenie procesu budowy rozwiązań systemów automatyki, ale też przekłada się na jakość tworzonych narzędzi w zakresie architektury funkcjonalnej i programowej.

2.6. Symulacja pomiarowych systemów sieciowych

Urządzenia pomiarowe, komunikujące się poprzez sieć komputerową zrewolucjonizowały budowy i wymusiły dynamiczny rozwój nowoczesnych systemów pomiarowych. Trudno właściwie wyobrazić sobie dzisiaj rozwiązania klasy BMS nie pozwalające na ciągłą komunikację z urządzeniami pomiarowo wykonawczymi. Wymuszony rozwój tego medium komunikacyjnego przyczynił się do poprawy niezawodności oraz szybkości przesyłanych danych. Infrastruktura kablowa stała się ważnym elementem organizującym łączność pomiędzy komputerami oraz urządzeniami zdalnymi (umieszczonymi w różnych lokalizacjach, miastach, krajach i kontynentach). Istotną cechą komunikacji stało się bezpieczeństwo i bezawaryjne działania sieci. Ponieważ budowa dużych sieci pomiarowo-wykonawczych jest zbyt droga, aby tworzyć je tylko na potrzeby eksperymentów, powszechnie korzysta się metod modelowania i symulacji pozwalających na tworzeniu substytutu zamodelowanej (rzeczywistej) sieci w symulacyjnym środowisku komputerowym. Taka forma analizy pozwala na obserwacje w warunkach zbliżonych do rzeczywistych, np. poprzez wypełnienie jej ruchem sieciowym podobnym do rzeczywistego, oraz weryfikację wad i zalet nowych rozwiązań i ich kompatybilności ze starszymi urządzeniami i standardami. Zatem symulacja komputerowa daje możliwość przetestowania nowych idei w rozmaitych warunkach (poprzedzających fizyczną realizację), jak również rekonstrukcji zdarzeń zachodzących w rzeczywistych sieciach. Istnieje przy tym możliwość odtworzenia i dokładnego zbadania zjawisk w charakterze ruchu sieciowego oraz topologii sieci bez konieczności ingerowania w rzeczywistą realizację [6, 53].

Klasyczna definicja symulacji komputerowej analizowanej sieci komputerowej sformułowana w [6, 53], jako proces, który odtwarza działanie pewnego układu w pewnym abstrakcyjnym środowisku. Zasadniczym elementem tego procesu są modele (matematyczne), które definiują elementy systemu oraz interakcję pomiędzy nimi w sposób zrozumiały dla środowiska symulacyjnego. Oczywiście model w większości przypadków ma charakter przybliżony, odwzorowując tym samym działanie modelowanego układu z pewną dokładnością. Jeżeli modelowany system jest zbyt skomplikowany aby, dało się uzyskać rozwiązanie analityczne, wymagane jest zastosowanie przybliżeń (np. wartości poszukiwanych parametrów). W symulacji komputerowej istotną rolę pełnią zmienne

stanu, przechowujące wartości najważniejszych (z punktu widzenia dynamiki) wielkości opisujących aktualny stan systemu symulowanego [6, 53].

Symulacja wymaga posługiwania się kilkoma pojęciami opisującymi czas oraz jego wpływ na reakcję symulowanego procesu. Wyróżniamy czas modelowy, który definiuje czas upływający w fizycznym procesie podlegającym symulacji. Czas symulowany, stanowi abstrakcyjną formę wielkości używanej w symulatorze do reprezentowania upływu czasu fizycznego. Inną formą czasu w symulacji komputerowej jest czas (zegarowy), który odnosi się do rzeczywistego czasu upływającego podczas programu symulacyjnego.

Systemy symulacji komputerowej dotyczą ciągłego (zmiany stanu systemu w funkcji czasu modelu lub symulowanego) i dziedziny czasu dyskretnego (zmiany stanu systemu następują w określonych dyskretnych momentach czasu - pomiędzy nimi stan systemu nie ulega zmianie). W przypadku symulacji dyskretnej możemy też wyszczególnić dwie charakterystyczne grupy rozwiązań. Pierwszą z nich jest symulacja z krokiem czasowym, nazywana symulacją krokową (ang.*time-stepped*), w której używa czasu symulacyjnego podzielonego na sekwencję krótszych kroków czasowych o różnej długości, natomiast proces symulacji przesuwa się po każdym kroku wyznaczając wartości zmiennych stanu. Najczęściej ten rodzaj symulacji wykorzystywany jest w odniesieniu do systemów, realizujących przybliżoną symulację modelu ciągłego, w których równania różniczkowe zastępowane są różnicowymi [6, 53].

W symulacji sterowanej zdarzeniami, wartości zmiennych stanu nie muszą ulegać zmianie w każdym kroku czasowym. Mechanizm kontroli upływu czasu może zmieniać stan zegara czasu symulowanego nie o stały krok, ale w momentach, w których następuje zmiana wartości zmiennej stanu [6, 53].

Badanie polegające na przeprowadzaniu eksperymentu symulacyjnego powinno być zgodne z określonym schematem (eksperymentu). Poniżej przedstawiono ogólny sposób postępowania z podziałem na poszczególne kroki.

Początkowy krok symulacji wymaga sformułowania problemu oraz określenia, w jaki sposób model odzwierciedla rzeczywistość. Następnie należy określić mechanizmy rządzące działaniem fizycznego systemu podlegającego modelowaniu. Należy pamiętać, że tworzony proces symulacyjny ma charakter programu komputerowego, którego działanie determinuje algorytm komputerowy oparty na zmiennych stanu, informujący o zdarzeniach oraz zmiennej czas symulowanego. Wykonywanie programu polega na modyfikacji (ewolucji) zmiennych stanu i tworzeniu nowych zdarzeń. Kolejny krok stanowi walidacja oraz weryfikacja modeli symulacyjnych. Walidacja jest potwierdzeniem zasadności modelu, polegającym na wykazaniu, że model komputerowy w środowisku eksperymentu posiada zadowalający stopień dokładności względem przedmiotowego modelu. Etap weryfikacji jest związany ze sprawdzeniem poprawności przekształcenia modelu formalnego do programu komputerowego, tj. sprawdzenie poprawności zaimplementowanych założeń. W dalszej kolejności w procesie modelowania i symulacji pojawia się planowanie

i realizacja przebiegu eksperymentu. Ta faza dotyczy przede wszystkim określenia warunków początkowych oraz przewidywania wartości zmiennych decyzyjnych, dla których będzie przeprowadzana symulacja. Warto podkreślić, że aby uzyskać statystycznie wiarygodne wyniki należy przeprowadzić odpowiednie długą symulację lub więcej przebiegów dla każdej wartości zmiennej decyzyjnej. Ostatni etap dotyczy analizy uzyskanych wyników. W przypadku symulacji probabilistycznej wynikiem może być rozkład szukanych wartości (należy wówczas też stosować narzędzia statystyczne do oceny wyników, takich jak analiza wariancji, analiza spektralna czy testy istotności) [6, 53].

Interesującym aspektem w kontekście symulacji rozwiązań sieciowych jest możliwość opisu tychże rozwiązań w dedykowanym języku programowania. Jednym z przykładów jest rozwiązanie oparte o język NDL[94] [11], dające jest możliwość opisu struktury sieci komputerowej (w formie połączeń) oraz komponentów wchodzących w jej skład. Pozwala to na sformalizowany opis struktur sieciowych, prowadząc w konsekwencji wstęp do uporządkowanego i systematycznego procesu optymalizacji i rekonfiguracji sieci.

[94] ang. *Network Description Language*.

3

PLATFORMA SMO - KONCEPCJA SYSTEMU ROZPROSZONEGO

Rozdział stanowi studium związane z genezą oraz realizowalnością platformy SMO. Dokonuje się przeglądu dostępnych funkcji. Przedstawia się szczegółowe założenia dotyczące budowanej platformy SMO. Prezentuje się sposoby rozwiązywania problemów sprzętowych i wydajnościowych opracowanego prototypu. Jako uzupełnienie systemu SMO opisuje się diagnostyczną szynę danych (DSB), która powstała z wykorzystaniem koncepcji ESB (Enterprise Service Bus).

3.1. Wprowadzenie

Sztuka obserwacji jest dyscypliną dojrzałą, od dawna rozwijaną i wzbogacaną przez pokolenia podróżników, astronomów oraz inżynierów. Przez wieki ewolucji człowiek wykształcił w sobie umiejętność prognozowania pewnych zachowań na podstawie obserwacji określonych symptomów. Początkowo były to bardzo oczywiste kwestie, związane z czynnikami destruktywnie wpływającymi na człowieka. Z czasem jednak, kiedy wzrósł poziom wiedzy o złożonych problemach świata rzeczywistego (fizycznego), wnikliwa obserwacja i analiza zjawisk zaowocowała możliwością przewidywania ich ewolucji służącego rozwiązywaniu rozmaitych problemów teoretycznych i praktycznych.

Monitoring to współczesna i dobrze posadowiona w technice wersja sztuki obserwacji, która jest aktualnie poparta olbrzymią wiedzą rozwiniętą w zakresie nauk przyrodniczych i technicznych. Szczególny rozwój tej dyscypliny w ostatnich latach wiąże się z konstrukcją układów diagnostycznych w zakresie nadzorowania stanu pracy obiektów (np.

przemysłowych). Takie systemy pozwalają inżynierom na bieżące śledzenie wybranych parametrów monitorowanych procesów lub urządzeń. Istnieje możliwość dokonywania zapisów (rekordów) w celu ich bieżącej (lub późniejszej) analizy oraz przewidywania niepożądanych zachowań lub potencjalnie niebezpiecznych stanów (usterek, uszkodzeń lub awarii). Rozwój technologii internetowych sprawił, że powstały też możliwości wykorzystania globalnej sieci w systemach monitorujących, co prowadzi do nieosiągalnej dotąd funkcjonalności, polegającej na analizie stanu urządzeń w korporacjach posiadających fabryki rozrzucone po całym świecie. Grupy ekspertów, rezydujących gdziekolwiek (np. w różnych krajach) uzyskują w ten sposób możliwość ciągłego monitorowania odległych i rozległych obiektów oraz podejmowania stosownych i skoordynowanych działań na podstawie wykrytych symptomów. Tego typu platformy diagnostyczne opierają się na pracy zespołu zintegrowanych systemów, łączącego technologie internetowe, wbudowane systemy mikroelektroniczne, systemy bazodanowe oraz dedykowane inteligentne systemy wnioskujące. Inwestowanie w rozwój takich systemów jest istotne, gdy występuje niebezpieczeństwo awarii skutkujących długimi przestojami produkcyjnymi lub innego rodzaju stratami (np. ekologicznymi lub finansowymi).

W niniejszej rozdziale przedstawione zostaną założenia projektowe i sposób realizacji inteligentnego (komputerowego) systemu do monitorowania obiektów przemysłowych z wykorzystaniem globalnej sieci oraz technologii bezprzewodowej transmisji danych [54].

3.2. Sieciowy Monitor Obiektu dla systemów rozproszonych

Sieciowy Monitor Obiektu (SMO) jest wynikiem kilkuletnich prac realizowanych w Katedrze Systemów Decyzyjnych i Robotyki (WETI) Politechniki Gdańskiej. Koncepcja SMO[1] stanowi aplikację sprzętowo-programową, która wykorzystuje najnowsze technologie z zakresu elektroniki, telekomunikacji, informatyki i automatyki do systemowej realizacji zadania, jakim jest diagnostyka i monitoring obiektów przemysłowych. W najprostszy sposób SMO można opisać jako system integrujący urządzenia pomiarowe z oprogramowaniem w celu zamierzonej obróbki danych, polegającej na analizie danych, wnioskowaniu, etc., aż do wytworzenia sygnału sterującego (sprzężenia zwrotnego), uwzględniającego nową sytuację. Może to być wykorzystane przy rekonfiguracji układu w systemach tolerujących usterki [56] lub przy metodach kompensacji anomalii lub defektu, jaki może pojawić się w monitorowanym procesie. Oryginalną koncepcję działania systemu przedstawiono w pracy [55]. Obecna realizacja systemu uwzględnia bezprzewodową komunikację z urządzeniami pomiarowymi, podejmowanie decyzji w środowisku wnioskowania

[1]SMO - Sieciowy Monitor Obiektu (ang. *System of Monitoring Objects via Network*)

oraz delegowania zadań do końcowych urządzeń wykonawczych. W niniejszej rozdziale przedstawione zostaną również nowe koncepcje dalszego rozwoju projektu SMO [57].

3.2.1. Koncepcja systemu zarządzania inteligentnymi obiektami

Podczas projektowania aplikacji rozważono kilka metod integracji różnych technologii w celu budowy spójnego i niezawodnego systemu. Realizacja projektu została podzielona na dwa główne etapy: pierwszy dotyczył budowy części sprzętowej, natomiast drugi związany był z budową aplikacji programowej dającej użytkownikowi dostęp do pożądanej funkcjonalności z przejrzystym interfejsem użytkownika. Modułowa struktura pozwala na tworzenie dowolnych konfiguracji systemu, zaś projektant zyskuje możliwość decydowania o szczegółach funkcjonalności aplikacji. Łatwość w obsłudze, automatyczne mechanizmy tworzenia kopii zapasowych oraz bezpieczeństwo przechowywanych informacji w bazach danych uznano za priorytetowe. Informacje o stanie dużych korporacji mają wartość strategiczną, gdyż każde awaryjne wstrzymanie produkcji może wpłynąć na spadek wartości notowań firmy, oraz pozbawić ją kolejnych zleceń. Istotną rolę wśród założeń projektowych odegrały mechanizmy diagnostyczne wspierające stały nadzór nad obiektami przemysłowymi różnych kategorii [54].

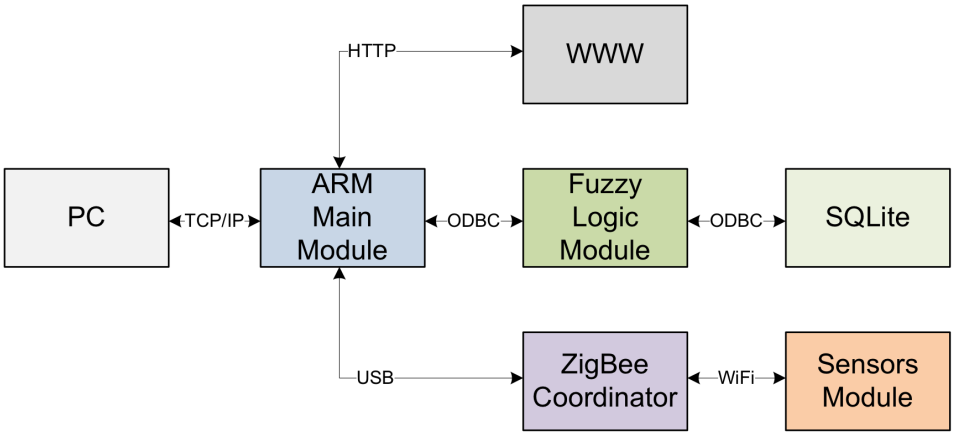
3.2.2. Opracowane rozwiązanie sprzętowe

Jądro sprzętowej części stanowi moduł sterujący pracą urządzeń pomiarowych i wykonawczych oparty na 32-bitowym mikrokontrolerze ARM[2] [7]. Jest on kontrolowany przez (specjalnie opracowany dla tego zadania) system operacyjny posadowiony na (darmowej) platformie Linux, który został zoptymalizowany do zadań SMO oraz wyposażony w narzędzia programowe wspomagające (funkcjonalną i bezawaryjną) pracę aplikacji. Informacje diagnostyczne wizualizowane są za pośrednictwem strony internetowej generowanej dynamiczne przy użyciu języków skryptowych. W skład opracowanej platformy wchodzi również moduł koordynujący ZDM-A1281-PN[3] (ang. *ZigBee Coordinator*) przedstawiony na rys. 3.1 oraz 3.2. Głównym zadaniem modułu jest nadzorowanie przesyłanych pakietów w sieci ZigBee oraz podłączonych do niej urządzeń (dodanie nowych urządzeń do sieci ZigBee wymaga udziału modułu koordynującego). Moduł ten łączy się poprzez magistralę USB z modułem sterującym w celu wymiany informacji pomiarowych i sterujących [3].

W skład części sprzętowej wchodzą również specjalizowane moduły pomiarowe. Pomiędzy lokalnymi modułami do transmisji bezprzewodowej a układami pomiarowymi zaimplementowano mechanizmy szybkiej i odpornej na zakłócenia wymiany danych w

[2]ang. *Advanced RISC Machine to 32-bitowa architektura procesorów RISC.*
[3]http://www.farnell.com/datasheets/68434.pdf

oparciu o interfejs I2C[4]. Metodę przesyłania informacji oraz mechanizmy służące do realizacji transmisji przedstawiono na rys. 3.1. Aplikacja umożliwia monitorowanie parametrów fizycznych procesów obiektowych. System wyposażony może być także w układy wykonawcze, które służą do sterowania pracą urządzeń zewnętrznych.



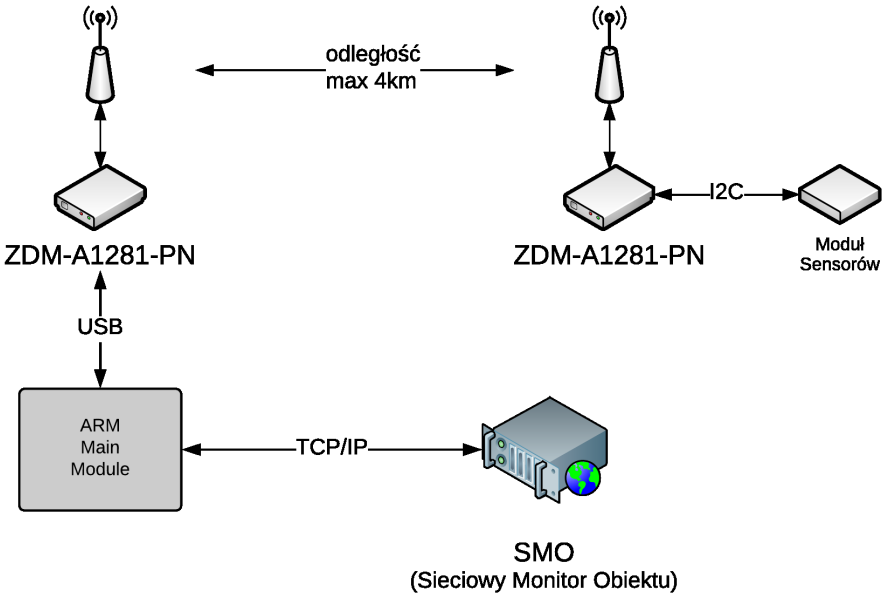
Rys. 3.1. Protokoły komunikacyjne (TCP/IP, HTTP, USB, I2C), elementy sprzętowe (ARM, PC, ZigBee) oraz typy kompatybilnego oprogramowania tworzącego strukturę SMO.

Bloki transmisji bezprzewodowej powstały w oparciu o mikrokontrolery ATmega[5], często wykorzystywane przy budowie systemów autonomicznych. Kontrolę nad układami AT pełni (darmowy) mikrosystem operacyjny TinyOS[6], który udostępnia substytut pracy wielozdaniowej (z zastosowaniem mechanizmu przerwań). Dzięki wymianie informacji pomiędzy poszczególnymi blokami bezprzewodowymi można prowadzić kontrolę zarówno układów wykonawczych jak i parametrów diagnostycznych. Konfigurację części sprzętowej SMO przedstawia rys. 3.2.

3.2.3. SMO aplikacja programowa

Mikrokomputerowa warstwa sprzętowa wymaga odpowiednio oprogramowania. W trakcie prac nad budową aplikacji zadbano o funkcjonalne dostosowanie wykorzystywanego sprzętu do przewidywanych narzędzi i środowisk programowych, z uwzględnieniem mechanizmów pozwalających na szybką modyfikację (funkcjonalności) programu. W programowaniu zastosowano rozwiązania pozwalające zmieniać funkcje systemu w czasie rzeczywistym - np. dokonywać okresowej rekonfiguracji programowych modułów zewnętrznym

[4]I2C - szeregowa, dwukierunkowa magistrala służąca do przesyłania danych pomiędzy urządzeniami elektronicznymi.
[5]ATmega - 8-bitowy mikrokontroler o niskim poborze mocy.
[6]TinyOS - system operacyjny urządzeń przenośnych, http://www.tinyos.net.



Rys. 3.2. Warstwa sprzętowa aplikacji SMO.

przyciskiem wyzwalającym. Opcje tego rodzaju są szczególnie użyteczne przy instalowaniu monitora SMO w nowych warunkach.

Warstwa programowa zawiera rozwiązania oparte na różnych językach programowania. Funkcjonalność modułu sensorów zrealizowano na mikroprocesorze ATmega, który to komunikując się za pośrednictwem interfejsu I2C z czujnikami, przekazuje pomiary do modułu bazodanowego [16, 52]. Moduły Zigbee (ZDM-A1281-PN), przestawione na rys. 3.2, wyposażono w narzędzia diagnostyczne pozwalające na sprawdzenie poprawności transmisji i jakości odbieranego sygnału oraz na generowanie alarmów wynikających z niepoprawnego trybu pracy. Ponieważ pomiędzy modułami bezprzewodowymi a układem zarządzającym może dojść do zerwania transmisji, wprowadzono mechanizm informowania o błędach oparty na diodach elektroluminescencyjnych. Zastosowane oprogramowanie pozwala na przydzielanie modułom ZigBee adresów w sieci bezprzewodowej, które mogą też być zmieniane za pośrednictwem stosownego oprogramowania [52].

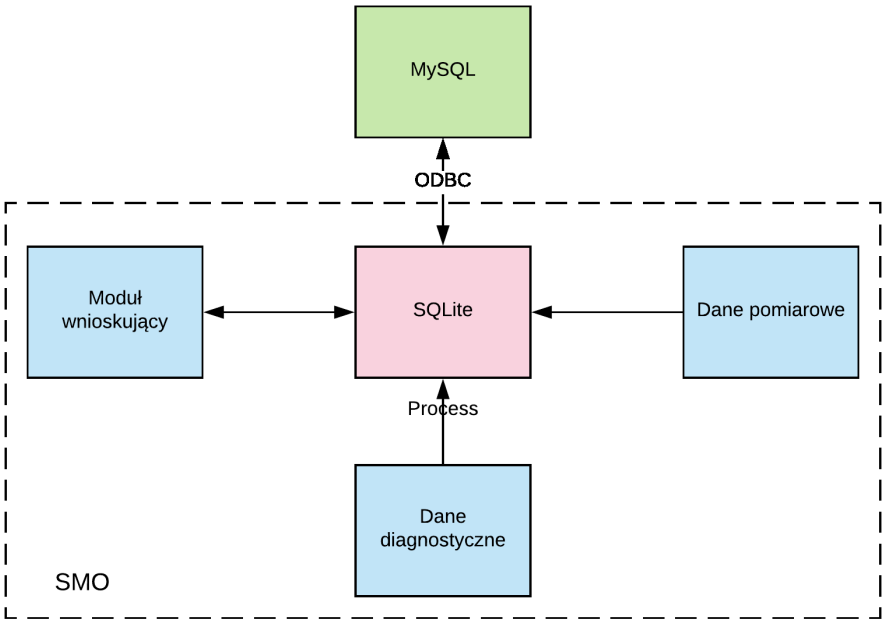
Koordynacją sieci zajmuje się moduł zarządzający ARM (ang. *Main Module*, rys. 3.2). Do jego zadań należy między innymi generowanie (testowych) zapytań do poszczególnych modułów bezprzewodowych w celu sprawdzenia, czy aktualnie komunikacja przebiega poprawnie. Aplikacja posiada możliwość zdefiniowania parametrów przesyłania danych (kolejności i odstępów czasowych) z poszczególnych modułów ZigBee. Umożliwia to realizację mechanizmu równomiernego obciążenia modułu przyjmującego dane (jak również całej sieci bezprzewodowej). Ważną opcję stanowi możliwość podłączenia

urządzenia koordynującego do sieci lokalnej LAN, przesyłającej pakiety według protokołu TCP/IP. Pozwala to na dostęp do przechowywanych danych z dowolnego miejsca oraz umożliwia (poprzez odpowiednie narzędzia) zdalne logowanie się do aplikacji oraz koordynowanie działania uruchomionych programów.

3.2.4. System zarządzania danymi

Przy realizacji projektu SMO przyjęto, że do uzyskania pełnej funkcjonalności potrzebne są dwa typy baz danych.

Bazę SQLite[7] zastosowano z myślą o module ARM, możliwej do zaimplementowania w urządzeniach przenośnych. Głównym zadaniem takiej bazy jest przechowywanie informacji modułu zarządzającego siecią (przychodzących z urządzeń pomiarowych). Moduł ARM jest urządzeniem autonomicznym służącym do analizy danych pomiarowych zawartych w bazie SQLite. Zwykle stosuje się w nim wnioskowanie rozmyte (opisane w paragrafie 3.2.5), przy czym funkcje przynależności niezbędne do realizacji takiego zadania również zapisuje się w tej bazie. Dzięki temu komunikacja z modułem realizowana jest w języku zapytań SQL[8] [52].



Rys. 3.3. Wymiana danych pomiędzy modułem SMO a zewnętrzną bazą danych.

[7]https://www.sqlite.org/index.html
[8]ang. *Structured Query Language (tj. strukturalny język zapytań).*

W trakcie rozwoju systemu SMO, powstał pomysł na integrację systemu wnioskującego opartego na module ARM z zewnętrznym serwerem bazodanowym (MySQL[9]). Dzięki tej konfigurację moduły SMO mogą być dowolnie rekonfigurowane lub wzbogacane o nowe reguły zachowań, określające sposób podejmowania decyzji jako interakcji na zmieniające się uwarunkowania (parametry wejściowe). Do realizacji połączenia wykorzystano interfejs ODBC[10], umożliwiający skryptom na tworzenie połączeń pomiędzy bazami danych. Na rys. 3.3 przedstawiono schemat połączeń pomiędzy wykorzystanymi bazami [52].

3.2.5. Moduł wnioskujący

Budowa systemu o dużym stopniu autonomizacji, który większość decyzji podejmuje samoczynnie, było jednym z głównych celów projektu SMO. Pracę rozpoczęto od bazodanowego systemu zbierania danych pomiarowych pozyskiwanych z czujników, w którym mechanizm zapisu i przechowywania pomiarów pozwala na szybkie dotarcie do informacji z wykorzystaniem języka zapytań SQL. Możliwość uzyskania natychmiastowej odpowiedzi na zadane do bazie zapytanie ułatwia śledzenie parametrów w czasie rzeczywistym. Zgodnie z założeniami, moduł wnioskujący pozwala na definiowanie oraz edytowanie reguł określających automatyczną reakcje na zachowania monitorowanego obiektu (zgodnie z posiadaną wiedzą ekspercką). Najbardziej efektywnym rozwiązaniem okazał się mechanizmu wnioskowania rozmytego. W bazie danych powstały tabele zawierające zdyskretyzowane funkcje rozmytej przynależności, determinujące również zmienne lingwistyczne [52, 25]. W oparciu o system zapytań SQL zaimplementowano bazę reguł, służącą do uzyskiwania informacji o stopniu spełnienia kryteriów wyboru. Proces wnioskowania jest wizualizowany, co pozwala na śledzenie poprawności działania systemu. W sytuacjach awaryjnych istnieje możliwość wyłączenia modułu automatycznego podejmowania decyzji oraz przejścia na manualne sterowanie urządzeniami (podłączonymi do bloków wykonawczych) przez operatora.

3.2.6. Platforma internetowa w SMO

Dostępność systemu z dowolnego miejsca jest ważną cechą budowanej aplikacji. Należy jednak wziąć pod uwagę aspekt udostępniania określonych informacji, często bezpośrednio związanych z procesem produkcyjnym. Do ochrony danych stworzono mechanizm autoryzowanego dostępu, pozwalający tylko określonym użytkownikom na logowanie się do aplikacji.

Zadaniem platformy internetowej jest też zapewnianie komunikacji pomiędzy eks-

[9]https://www.mysql.com/
[10]ang. *Open DataBase Connectivity, tj. otwarte połączenie baz danych.*

pertem a grupami pracowników odpowiedzialnych za utrzymanie ruchu. Ekspert jest użytkownikiem uprzywilejowanym, z dostępem do wszystkich informacji diagnostycznych zawartych w systemie. Ma on również możliwość wizualizacji tych danych, eksportu do znanych formatów biurowych oraz generowania zadań dla pracowników obsługujących proces (również w celu zapobiegania lub usuwania awarii) [52].

Opracowana w ten sposób aplikacja pozwala ekspertom na koordynowanie wielu obiektów w tym samym czasie oraz zbiorczą analizę danych i bieżące interwencje.

3.3. Koncepcja systemu do zarządzania inteligentnymi obiektami

Rozproszone, inteligentne systemy pomiarowe rozpowszechniają się coraz bardziej. Koncepcja monitowania wielu elementów składowych systemu pozwala też na projektowanie i budowanie użytecznych narzędzi służących analizie predykcyjnej w czasie rzeczywistym. Systemy tej klasy pozwalają nie tylko na wykrycie problemu, ale również na wprowadzenie pewnego wyprzedzenia czasowego przy informowaniu o zbliżającej się awarii.

W niniejszym podrozdziale przedstawimy i scharakteryzujemy poszczególne elementy składowe platformy SMO – rozproszonego systemu do monitorowania i zarządzania inteligentnymi systemami i obiektami.

3.3.1. Rola systemu diagnostyki i sterowania obiektami

Wykorzystanie systemów takich jak SMO w realizacji zadań diagnostycznych w inteligentnym budownictwie wiąże się z poszukiwaniem optymalnego podejścia do organizacji procesu zarządzania i konserwacji budynków stanowiących złożone infrastrukturalnie miejsce pobytu, pracy lub zamieszkania.

Platforma SMO stanowi uniwersalne rozwiązanie, które można modyfikować (rozbudowywać) korzystając z otwartego protokołu programowania. Diagnostyka w takich systemach pełni niezwykle istotną rolę, ponieważ we współczesnej infrastrukturze powszechnie stosuje się systemy automatycznego sterowania. Bezusterkowa praca takich systemów oparta jest na ciągłej konserwacji i serwisie. Zadaniem systemu SMO jest bieżąca analiza i wydobywanie informacji ukrytych w danych w celu optymalizacji modelowanego procesu (np. minimalizacja liczby awarii lub przestojów wynikających z problemów sprzętowych). Platforma pozwala na zastosowanie modeli oraz dokonywanie predykcji służących ostrzeganiu o zbliżających się nieprawidłowościach [30]. Ogólna koncepcja opracowania zautomatyzowanego procesu wykrywania błędów i diagnostyki (FDD) przedstawiona została w rozdziale 2.3.

3.3.2. Relacja pomiędzy poszczególnymi elementami systemu

Podczas projektowania złożonych systemów sprzętowych mogą pojawić się trudności związane z komunikacją pomiędzy warstwą komputerowej aplikacji a układami wykonawczymi. W omawianym przypadku dużym wyzwaniem było połączenie bazodanowego podsystemu Sieciowego Monitora Obiektu z układem do zarządzania siecią bezprzewodowych czujników pomiarowych. Główna trudność związana była z budową bezpiecznego systemu komunikacji aplikacji internetowej z urządzeniami. Skutecznym rozwiązaniem okazało się połączenie oparte na mechanizmie applet-servlet[11], możliwym od realizacji w języku Java.

Obsługa dwóch niezależnych baz danych (MySQL i SQLite) w systemie SMO implikuje konieczność budowy wspólnego efektywnego mechanizmu tworzenia kopii zapasowych. Realizację tego zadania oparto na języku PERL[12], środowisku skryptowym BASH[13], oraz mechanizmach dostępnych w systemie Linux o nazwie CRONTAB[14].

3.3.3. Diagnostyczna Szyna Danych (DSB)

W informatyce dawno dostrzeżono problem związany z łączeniem wielu różnych aplikacji (często pisanych w różnych językach, z użyciem rozmaitych technologii). Podejściem pozwalającym na rozwiązanie problemów integracji różnych środowisk programowania okazała się koncepcja określana mianem architektury zorientowanej usługowo. Jednak SOA[15] to jedynie pomysł na budowanie systemów zorientowanych na dostarczaniu usług, działających według określonych kryteriów, niezależnie od konkretnych technologii [57, 22]. Takie uniezależnienie od systemu operacyjnego oraz języka programowania pozwala na ich wykorzystywanie w środowiskach heterogenicznych [10]. Dla użytkownika końcowego serwisy są czarnymi skrzynkami, których zawartość nie musi być powszechnie dostępna; liczy się jedynie świadomość podawania parametrów wejściowych i otrzymywania danych przetworzonych na wyjściu.

W klasycznej architekturze SOA, usługi powinny posiadać następujące właściwości (choć w praktyce nie zawsze wszystkie są spełnione):

- być hermetycznym bytem, używanym niezależnie od innych,
- dostępne w sieci,
- posiadać zdefiniowany interfejs,

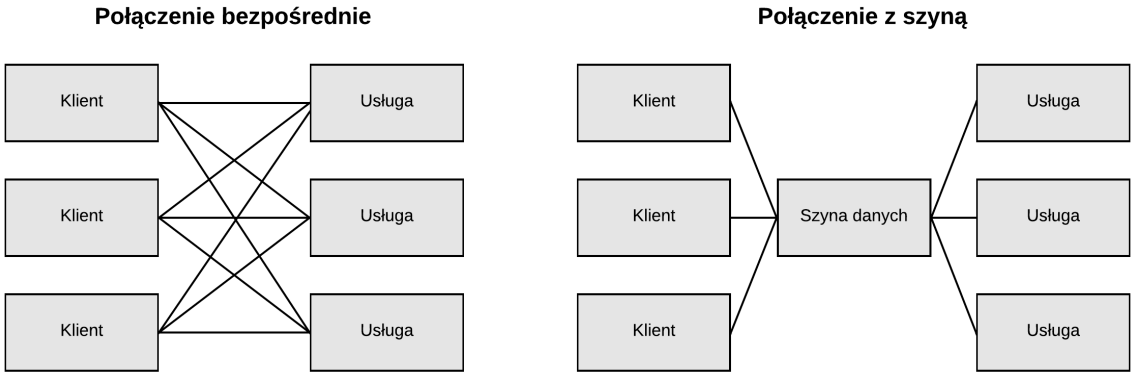
[11] applet-servlet - model komunikacji, zawarty w mechanizmach języka Java, który działa po stronie serwera w modelu żądanie-odpowiedź.
[12] PERL - interpretowany język skryptowy przeznaczony głównie do pracy nad danymi tekstowymi.
[13] BASH - powłoka systemowa UNIX napisana dla projektu GNU.
[14] CRONTAB - aplikacja przerwaniowa zajmująca się okresowym wywołaniem innych programów.
[15] ang. *Service Oriented Architecture.*

- być niezależne od platformy.

Do korzystania z usługi nie jest zatem potrzebna wiedza na temat szczegółów implementacji usługi, a jedynie znajomość interfejsu (usługa może być realizowana w dowolnym języku programowania oraz uruchamiana w różnych systemach operacyjnych) [10].

Warto tu przytoczyć sposób sterowania zdarzeniami z wykorzystaniem architektury SOA. Zdarzenia wpływające na normalny przebieg procesu mogą występować w dowolnej kolejności i dowolnym momencie. Aplikacje wymieniające dane (w zautomatyzowanych procesach) komunikują się ze sobą za pomocą sterowanej zdarzeniami architektury SOA (co pozwala na elastyczne reagowanie na zmieniając się wymagania i warunki procesowe). Usługi udostępniają proste interfejsy, pozwalające na ich asynchroniczne współużytkowanie. Umożliwia to łączenie ze sobą usług w celu tworzenia zespołów usług (definiowanie bardziej złożonych procesów).

ESB[16] stanowi implementację warstwy pośredniej zgodnej z paradygmatem SOA, gdzie szczególny nacisk położony został na łączenie usług i metod transportu informacji pomiędzy nimi [10]. Model architektury ESB zakłada wykorzystanie centralnej szyny danych do połączeń pomiędzy usługami. Wszelkie wiadomości przesyłane pomiędzy usługami przekazywane są za pośrednictwem szyny. Zaletą takiej konfiguracji jest mała liczba połączeń pomiędzy usługami w stosunku do typowych konfiguracji wymagających nawiązywania bezpośredniego połączenia. W przypadku n usług, liczba połączeń z wykorzystaniem ESB wynosi n, w porównaniu z $\frac{n(n-1)}{2}$ w przypadku połączeń bezpośrednich. Różnice pomiędzy połączeniami przedstawione zostały na rys. 4.1.



Rys. 3.4. Komunikacja pomiędzy serwisami w modelu ESB [10].

Koncepcja ESB zakłada wykorzystanie jednego spójnego modelu danych. Nie oznacza to jednak, że serwisy działające w oparciu o szynę muszą zapewniać taki sam model

[16]ang. *Enterprise Service Bus.*

przesyłanych informacji (wymaganie trudne do spełnienia w rzeczywistych warunkach, często uniemożliwiające integrację). W ESB rozwiązaniem tego problemu jest wykorzystanie adapterów, które umieszcza się pomiędzy serwisami a szyną, w celu zapewnienia konwersji pomiędzy poszczególnymi modelami danych [10].

Szyna nie jest jedynie warstwą transportową dla przesyłanych wiadomości pomiędzy usługami. Poza charakteryzuje się takimi funkcjami jak:

- transformacja danych na inny format,
- zarządzanie wieloma modelami tranzakcji i bezpieczeństwa (łączy różne modele integrowanych usług),
- agregacja żądań dostępu do usług,
- obsługa protokołów sieciowych z zachowaniem jakości usług (QoS[17]) [10].

W architekturze ESB wyróżnia się kilka metod, według których dokonywana jest transformacja przesyłanych komunikatów pomiędzy usługami. Metody te można dowolnie łączyć ze sobą, tworząc złożone łańcuchy metod [8, 10]:

1. Wzbogacenie (ang. *enrichment*) - zawartości wiadomości poprzez dołączenie dodatkowych parametrów (pochodzących np. z bazy danych),
2. Przełaczanie protokołu (ang. *protocol switch*) - zmiana protokołu komunikacji pomiędzy usługodawcą a usługobiorcą (klientem), które może mieć miejsce w dowolnym punkcie na drodze pomiędzy współpracującymi stronami,
3. Transformacja (ang. *transformation*) - zmiana formatu wiadomości z postaci rozumianej przez usługobiorcę do formatu akceptowalnego przez usługodawcę, która może mieć formę enkapsulacji, dekapsulacji, szyfrowania itd.,
4. Przekierowanie (ang. *routing*) - wybór usługodawcy na podstawie z góry określonych kryteriów, które mogą być rozpatrywane na podstawie zawartości komunikatu, kontekstu lub cech dostępnych usługodawców,
5. Dystrybcuja (ang. *distribution*) - pozwala na przesyłanie wiadomości do wielu aplikacji klienckich zainteresowanych usługami (działa na zasadzie modelu subskrypcyjnego),
6. Monitoring - umożliwia analizę przepływających wiadomości w celu logowania, pomiaru wykorzystania zasobów lub rozwiązywania innych problemów, itd.,

[17] ang. *Quality Of Service.*

7. Korelacja - pozwala na definiowanie reguł identyfikacji i rozpoznawania wzorców w komunikatach.

Istotnym elementem podnoszącym możliwości konfiguracyjne rozwiązania ESB są wzorce rozlokowania (ang. *deployment patterns*). Idee wzorca pozwalające na używanie szyny w różnych konfiguracjach (nie tylko w formie centralnego punktu integracyjnego) przedstawiono poniżej [8, 10]:

1. Global ESB - jedna wspólna szyna, gdzie każdy dostawca usług jest widoczny dla każdego klienta poprzez heterogeniczne, centralnie zarządzane środowisko,

2. Directly connected ESB - niezależne instancje ESB posiadające wspólny rejestr usług,

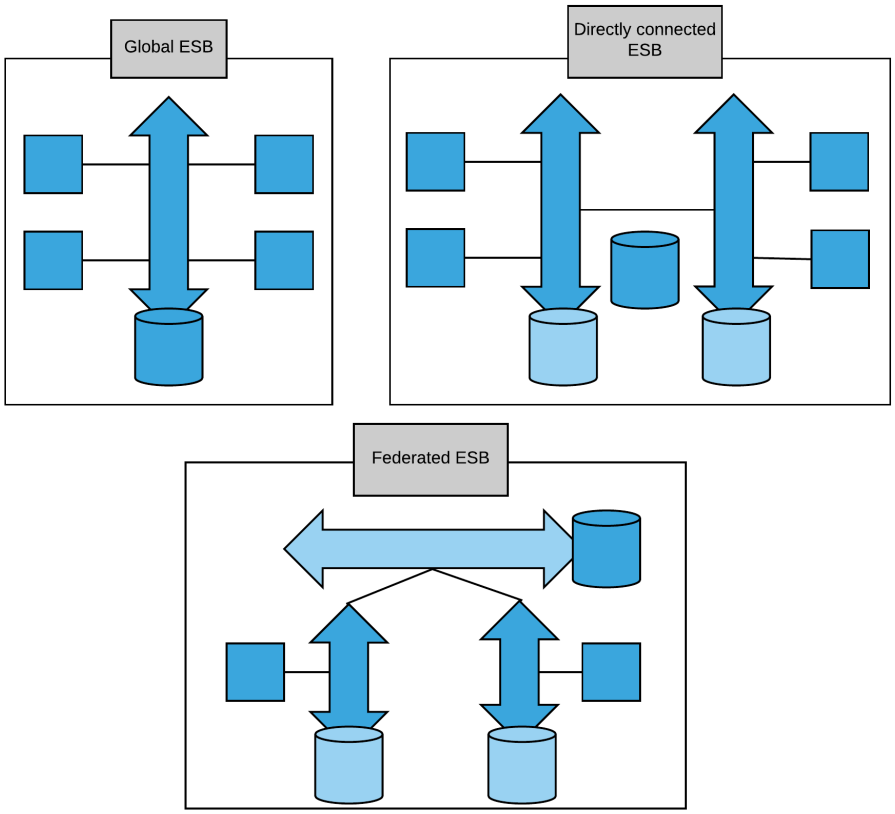
3. Federated ESB - pozwala na zdefiniowanie jednego nadrzędnego ESB, który łączy zależne od niego instancje (usługodawcy i usługobiorcy łączą się do nadrzędnego lub podległego ESB w celu uzyskania dostępu do usług)

Magistrala DSB, zaimplementowana w oparciu o architekturę ESB, oferuje rozproszoną szynę pozwalającą na integrację wielu protokołów komunikacyjnych jednocześnie. Aplikacje i komponenty integracji na poziomie abstrakcji są od siebie oddzielone. Łączone są natomiast za pośrednictwem magistrali jako logiczne punkty końcowe, prezentowane jako usługi sterowane zdarzeniami [12]. W przypadku zastosowania magistrali otrzymujemy rozwiązanie, które może obejmować zasięgiem swojego działania rozbudowane przedsiębiorstwo (wielooddziałowe), oferując globalny dostęp do wspólnego medium komunikacji oraz transformacji między procesowej. Taka forma architektury zapewnia cechujący się dużą elastycznością fundament, który pozawala na przystosowanie magistrali do dowolnego typu środowiska integracji [12].

Magistrala DSB udostępnia metody integracji, zapewniające podstawy luźno powiązanej sieci o wysokim stopniu rozproszenia. Pozwala to na osiągniecie większych możliwości skalowania niż w przypadku rozwiązań pracujących w konfiguracji gwiaździstej (EAI[18]). Magistrala stanowi platformę integracji, umożliwiającą przesyłanie komunikatów, transformację danych i inteligentne kierowanie ruchu danych w celu niezawodnego nawiązywania połączeń oraz koordynowania interakcji pomiędzy różnymi aplikacjami (gdzie różnorodność dotyczy poziomu technologii wykonania).

Realizację koncepcji SOA stanowi opracowany model aplikacji DSB, wykorzystywany pierwotnie do diagnostyki budynków. Pomysł na zastosowanie szyny danych w diagnostyce obiektów przemysłowych związany jest z możliwością użycia stworzonych wcześniej w różnych technologiach modułów programowych. W trakcie prac nad Diagnostyczną

[18]ang. *Enterprise Application Integration.*



Rys. 3.5. Wzorce rozlokowania w modelu ESB [10].

Szyną Danych przeanalizowano sposoby i metody realizacji innych systemów zorientowanych usługowo, opartych o ESB, tj. MULE[19] oraz Biztalk[20].

Stabilność oraz mnogość dostępnych funkcji w magistralach ESB systematycznie wzrasta. Przyczyniło się to do wyboru tej technologii jako narzędzia wspomagającego budowę rozproszonego systemu dostosowanego do potrzeb diagnostyki obiektów przemysłowych, wykorzystującego możliwości opracowanego wcześniej modułu SMO.

Szyna DSB pozwala na podłączenie nie tylko aplikacji komputerowych, ale również umożliwia obsługę urządzeń komunikujących się w standardzie TCP/IP[21] (w tym moduły SMO). Z punktu widzenia budowy i eksploatacji systemów pomiarowych, otrzymuje się w ten sposób rozwiązanie pozwalające na integrację czujników z systemami przetwarzania informacji pomiarowych. Wspólna szyna danych posiada jeden standard zapisu i przesyłania informacji. W przypadku rozwiązania DSB format ten jest zgodny ze stan-

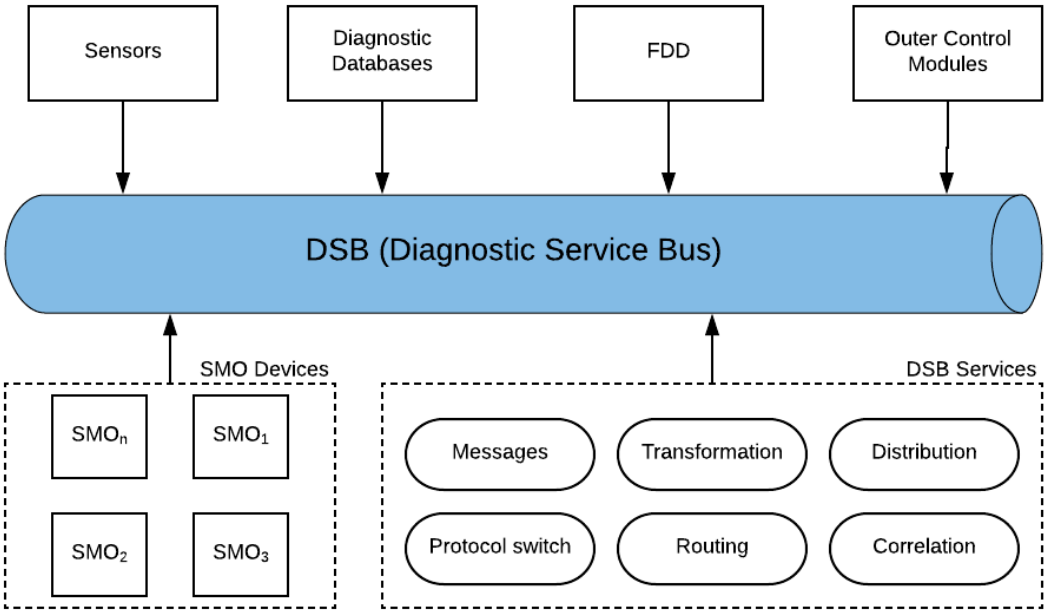
[19] aplikacja o architekturze zorientowanej na usługi.
[20] komercyjne rozwiązanie klasy ESB firmy Microsoft.
[21] ang. *Transmission Control Protocol/ Internet Protocol.*

dardem XML. Szyna DSB umożliwia dokonywanie wielokrotnego przetwarzania transportowanej informacji. Definiuje się ustalony kierunek przepływu informacji pomiędzy blokami transformującymi. Zastosowanie architektury zorientowanej na usługi umożliwia zatem podłączenie dowolnych modułów wspomagających procesy diagnostyczne w inteligentnych rozwiązaniach. Podczas prac badawczo-projektowych opracowano moduł FDD (2.3) oraz moduł klasyfikujący dane pomiarowe z wykorzystaniem naiwnego klasyfikatora *Bayesa*. Opracowany moduł pozwalał na przypisywanie wartości pomiaru, wzbogaconej o wektor dodatkowych informacji środowiskowych do odpowiedniej grupy. W ramach prac zdefiniowaliśmy kilka grup pozwalających na określenie czy monitorowany układ działa poprawnie czy też znajduje się w trybie sugerującym pojawiającą się awarię. Schemat szyny DSB wraz z opracowanymi modułami przedstawiono na rys. 3.6.

Sieciowy Monitor Obiektu [56, 57] jest aplikacją umożliwiającą autonomiczną obserwację i analizę monitorowanych wielkości fizycznych, która reaguje w sytuacjach zdefiniowanych przez użytkownika (eksperta) jako niepoprawne, bądź też krytyczne. Do analizy sygnałów pomiarowych stosuje się moduł wnioskowania (rozmytego). Testy oryginalnego systemu w warunkach rzeczywistych uwypukliły problem związany z przetwarzaniem dużej liczby danych pomiarowych (ograniczenia wydajnościowe platformy SMO wynikają z zastosowanej architektury sprzętowej a w szczególności modułów ZDM-A1281-PN). Wykorzystanie systemów wbudowanych wydłuża proces wnioskowania, przekraczając tym samym stawiane ograniczenia czasowe. Stąd też pojawiła się idea wykorzystania wspólnej szyny danych DSB dla urządzeń SMO, która umożliwia szybką wymianę informacji oraz otwiera nowe możliwości obliczeniowe (poza modułami SMO).

Początkowo magistrala DSB miała jedynie wspomagać medium komunikacyjne. Jednak w trakcie prac projektowych powstała koncepcja wykorzystania możliwości integracyjnych i obliczeniowych szyny. Dzięki temu w przypadku analizy danych, urządzenia mogą przekazywać je do magistrali w celu wykorzystania mocy obliczeniowej DSB i zwiększenia całkowitej wydajności systemu. Przepływ danych uzależniony jest od konfiguracji systemu. Nieprzetworzone dane są kolejkowane i oczekują na przetworzenie z wykorzystaniem szyny. Zaletą takiego rozwiązania jest możliwość wykorzystania tej samej informacji przez kilka aplikacji podłączonych do szyny DSB jednocześnie, np. moduł wnioskujący, raportujący czy wykonawczy. Zrównoleglenie operacji bezpośrednio przyczynia się do zwiększenia wydajności systemu diagnostycznego oraz kompleksowości realizowanych usług [57].

Powszechnym i dobrze znanym w automatyce rozwiązaniem służącym do komunikacji pomiędzy urządzeniami w sieciach przemysłowych jest protokół CANOpen. Pozwala on na standaryzację sposobu komunikacji pomiędzy urządzeniami w systemach wbudowanych. Opracowaniem oraz wdrożeniem tego rozwiązania zajęła się firma Bosch i między-



Rys. 3.6. Schemat szyny DSB.

narodowa grupa CiA[22] zrzeszająca użytkowników i producentów urządzeń. CANOpen charakteryzuje się wysoką niezawodnością i odpornością na zakłócenia. Niewątpliwym atutem takiego interfejsu są też stosunkowo niskie koszty wdrożenia oraz dostępność urządzeń wykorzystujących to podejście.

DSB w stosunku do CANOpen daje jeszcze większe możliwości. Informacja przekazywana szynie DSB, pozwala na przydzielanie jej priorytetów, tak jak ma to miejsce w CANOpen, ale istnieje też możliwość przetwarzania danych (przekazywanych przez magistralę) w tym samym czasie przez kilka procesów (aplikacji korzystających równolegle z DSB).

Rozwiązanie DSB-CANOpen pozwala na implementację struktur przetwarzania wielopoziomowego. Sygnały pomiarowe mogą być wówczas klasyfikowane z różnymi priorytetami ważności oraz adekwatnie przetwarzane. DSB posiada zaimplementowane stosowne protokoły komunikacyjne pozwalające na łatwą implementację nowych funkcji. Instalacja dodatkowego oprogramowania sprowadza się bowiem do połączenia zbudowanej aplikacji z szyną DSB w oparciu o odpowiedni standard komunikacyjny np. SOAP[23], warstwy

[22]ang. *CAN in Automation.*
[23]ang. *Simple Object Access Protocol.*

TCP/IP, JMS[24], JDBC[25], HTTP[26]. Funkcjonalność dotąd niezrealizowana będąca aktualnie przedmiotem badań, związana jest ze spełnieniem standardów stawianych systemom czasu rzeczywistego.

3.3.4. Wykorzystanie DSB w diagnostyce budynkowej

Biorąc pod uwagę zadanie projektowania rozwiązań diagnostycznych, wybór architektury ESB do budowy rozproszonych systemów agregujących i przetwarzających danych pochodzące z wielu źródeł wydaje się być w pełni uzasadniony. Za taką konfiguracją przemawia stosunkowo duża łatwość w integracji danych oraz możliwość tworzenia rozwiązań programowych komunikujących się przez wspólne medium.

3.4. Rozproszone systemy PDS

Rozproszone systemy pomiarowo-diagnostyczno-sterujące to kolejna generacja komputerowych systemów pomiarowych, w których obiekty i urządzenia rozmieszczone są w znacznych odległościach od siebie. Obszar zastosowania takich systemów jest bardzo szeroki i dotyczy takich obiektów jak przedsiębiorstwa infrastruktury komunalnej (wodociągi, sieci ciepłownicze i gazowe), duże budynki (szkoły, hotele, w których monitoruje się temperaturę i obecność dymu oraz inne parametry), przedsiębiorstwa przemysłowe i biznesowe, kopalnie, elektrownie, budynki korporacyjne czy też instytucje naukowe.

Kryterium dotyczące podziału systemów pomiarowych na rozproszone i nierozproszone nie jest dokładnie sprecyzowane. Umownie za rozproszone uważamy komputerowe systemy PDS, które mają urządzenia rozmieszczone w **większej liczbie miejsc** niż jedno oraz w których odległość pomiędzy przyrządami jest **większa niż długość przewodu** interfejsowego.

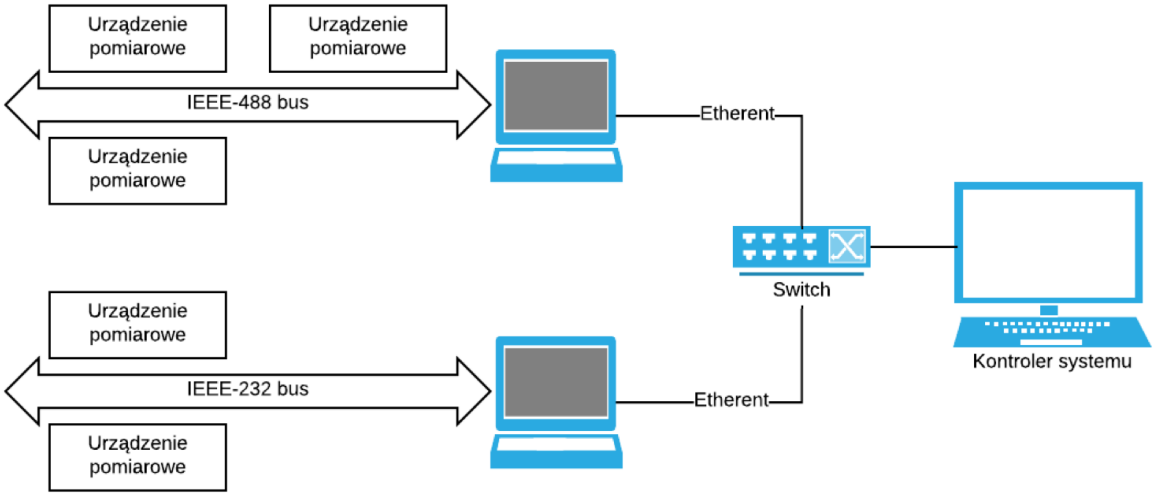
Kluczowy wpływ na rozwój rozproszonych systemów PDS miał dynamiczny rozkwit lokalnych sieci komputerowych LAN[27] oraz globalnych WAN[28] (Internet). To właśnie te grupy sieci są najczęściej wykorzystywane przy budowie rozległych sieci PDS (pomiarowo-diagnostycznych-sterujących).

Można wymienić trzy klasy systemów PDS wykorzystujących do komunikacji sieć LAN: hierarchiczne, natywne (LAN) oraz mieszane (konwerterowe).

Systemy hierarchiczne to sieci w których na najniższym poziomie istnieją podsystemy

[24]ang. *Java Messaging Service.*
[25]ang. *Java DataBase Connectivity.*
[26]ang. *Hypertext Transfer Protocol.*
[27]ang. *Local Area Network.*
[28]ang. *Wide Area Netowork.*

pomiarowe z odpowiednim interfejsem, np. IEEE-488[29] lub RS-232[30]. Niższą warstwę systemu może stanowić sieć LAN, do której dołączane są komputery-kontrolery podsystemów oraz kontroler systemu (w hierarchicznych systemach pomiarowych sieć typu LAN służy do typowej komunikacji pomiędzy komputerami, monitorującymi w sposób ciągły stan urządzeń pomiarowych). Sieć LAN wykorzystywana jest również do wymiany prostych informacji (danych i rozkazów). Schemat działania systemu hierarchicznego przedstawiono na rys. 3.7.



Rys. 3.7. Hierarchiczny system pomiarowy działający w sieci Ethernet [74].

Drugą klasą rozproszonych systemów pomiarowych są systemy natywne z wbudowanym interfejsem LAN. W grupie tej zarówno kontroler, jak i urządzenia pomiarowe są bezpośrednio dołączone do sieci komputerowej. Warto zaznaczyć, że do komunikacji z siecią nie wymagane są żadne dodatkowe urządzenia konwertujące, popularnie określane jako konwerter interfejsu.

Trzecią klasę systemów pomiarowych stanowią systemy mieszane wykorzystujące konwertery interfejsów: IEEE-488/LAN[31], RS-232C/LAN[32] lub RS-485/LAN[33]. Takie rozwiązania pozwalają na realizację komunikacji pomiędzy komputerami działającymi w sieci LAN (sygnały sterujące) a urządzeniami pomiarowymi podłączonymi do odpowiedniej magistrali (IEEE-488 lub systemu RS-232C) [74].

[29] https://en.wikipedia.org/wiki/IEEE-488
[30] https://en.wikipedia.org/wiki/RS-232
[31] http://sine.ni.com/nips/cds/view/p/lang/pl/nid/209210
[32] https://www.perle.com/products/rs232-to-ethernet.shtml
[33] http://www.usconverters.com/rs485-ethernet-converter

3.4.1. Implementacja rozproszonego środowiska seciowo-pomiarowego (PDS) na platformie SMO

System SMO opracowano i realizowano jako rozproszoną platformę pomiarowo-diagnostyczno-sterującą (PDS), w postaci uniwersalnej aplikacji, która zapewnia możliwość połączenia elektronicznego systemu (kontrola zadanych parametrów fizycznych) z internetową platformą służącą do komunikacji oraz analizy przepływających przez nią informacji. System projektowano z wykorzystaniem architektury modułowej, zarówno w kontekście programowym, jak również sprzętowym. Istotne z punktu widzenia użyteczności projektowanej aplikacji jest wyposażenie jej w urządzenia wykonawcze, pozwalające na sterowania niezależnymi obwodami.

Od początku prac prototypowych, realizowany projekt środowiska rozważany był w kategoriach narzędzia pracującego w trybie rozproszonym. Wielourządzeniowe sieci oparte na inteligentnych czujnikach pokrywających swoim działaniem duże obszary, stają się popularnymi rozwiązanymi wykorzystywanymi w zakładach przemysłowych, inteligentnych budynkach oraz fabrykach. Monitor SMO dzięki wykorzystaniu odpornej na zakłócenia technologii ZigBee/XBee dostosowywany był do pracy w warunkach trudno dostępnych, silnie zakłóconych, ale otwartych optycznie (bez przeszkód). Praca w takim środowisku możliwa była jedynie poprzez zastosowanie odpowiednich narzędzi do filtracji i wzmacniania transmitowanego sygnału. Zaletą okazała transmisja danych pomiędzy poszczególnymi węzłami sieci pomiarowej, wykorzystująca topologię gwiazdy oraz wzmocniony mechanizm kontroli przesyłanych informacji. Urządzenia działające w takiej sieci mogą być autonomiczne, a w przypadku awarii któregoś z węzłów w sieci pomiarowej, inne urządzenie może przejąć jego funkcje (zaś po dokonaniu rekonfiguracji sieci, system może powrócić do normalnego stanu pracy).

3.4.2. Ograniczenia platformy SMO

Opracowana koncepcja Sieciowego Monitora Obiektu ma ograniczenia, które warto brać pod uwagę podczas wdrażania go na środowisku produkcyjnym. Stworzona implementacja jest rozwiązaniem prototypowym, służącym w głównej mierze do przetestowania i weryfikacji założonych hipotez badawczych. Podczas tworzenia scenariuszy testowych weryfikujących działanie modułu przy komunikacji bezprzewodowej (ZigBee/XBee) zauważono duży problem w przesyłaniu komunikatów w środowisku o bardzo dużym zaszumieniu, a zwłaszcza w przestrzeniach zawierających przeszkody stałe (np. grube ściany, drzewa, liście itd.). Warto podkreślić, że zastosowane urządzenia nadawczo-odbiorcze posiadają stosunkowo małą moc (ZDM-A1281[34]) wyjściową na poziomie 3dBm. W celu realizacji komunikacji w trudnych przestrzeniach warto rozważyć rozszerzenie sieciowego

[34]http://www.farnell.com/datasheets/68434.pdf

monitora o komunikacyjny radiomodem np. (modele readiomodemów firmy SATEL[35].

Zastosowane układy AVR[36](ATMEGA-1281[37]) w opracowanym prototypie platformy SMO, z punktu widzenia wydajności obliczeniowej, również mogą stanowić pewne ograniczenie. Według założeń podstawowa funkcjonalność dotyczyła transmisji oraz komunikacji z czujnikami i blokami wykonawczymi. W przypadku realizacji bardziej złożonych zadań można zaproponować rozszerzenie możliwości obliczeniowych o bardziej zaawansowane jednostki obliczeniowe.

3.4.3. Wykorzystanie platformy w rozwiązaniach przemysłowych

Proces wdrożenia kompletnego systemu, rozpoczyna się od określenia podstawowych wymagań sprzętowych odpowiednich do monitorowanego obiektu. System musi być niezawodny, a czujniki pomiarowe odpowiednio skalibrowane tak, aby nie generowały błędnych odczytów. Ważnym etapem jest testowy okres obserwacji, w którym należy sprawdzić poprawność transmisji danych z czujników pomiarów, określić jakość przesyłania danych z wykorzystaniem technologii bezprzewodowej oraz zapoznać użytkowników z mechanizmami obsługi aplikacji. Istotnym elementem procesu wdrażania jest budowa kompletnej infrastruktury sieciowej umożliwiającej odpowiednie skonfigurowanie systemów automatycznego tworzenia kopii zapasowych oraz powiadamiania użytkowników o wszelkich odstępstwach od prawidłowej pracy.

3.4.4. Perspektywy dalszego rozwoju platformy - zdefiniowanie płaszczyzn rozwoju platformy SMO

Sieciowy Monitor Obiektu podlega nieustannemu procesowi rozbudowy i testowania tworzonych funkcjonalności. W planach jest budowa centralnej części systemu opartej na wydajnym mikroprocesorze z rodziny ARM. W fazie badawczej jest nowa, jeszcze bardziej zoptymalizowana wersja systemu operacyjnego pozwalającego na wydłużenie prace baterii zasilającej oraz jeszcze łatwiejszą integrację z szyną DSB. Rozważana jest również możliwość wykorzystania innych typów sensorów pomiarowych (np. czujników przepływu, detekcji ruchu, zbliżeniowych itd.) oraz rozbudowy toru nadawczo-odbiorczego wykorzystującego bezprzewodową technologie ZigBee/XBee [54].

[35]https://www.satel.com/products/radio-modems/
[36]https://en.wikipedia.org/wiki/Atmel_AVR
[37]http://www.microchip.com/wwwproducts/en/ATmega1281

3.5. Podsumowanie

Rozwój mobilności i technologii internetowych oraz globalizacja gospodarki światowej, uwzględniające kluczowe aspekty zarządzania przepływem informacji, otworzyły też zupełnie nowe możliwości opracowywania i konstruowania innowacyjnych rozwiązań technicznych.

Zbudowana aplikacja stanowi uniwersalne narzędzie, które może być zaadaptowane do różnego rodzaju zadań, wspomagających pracę ekspertów, obejmuje m.in. centrum zarządzania zadaniami, mechanizmy wizualizacji oraz inteligentne algorytmy decyzyjne. Skutecznej koordynacji działań diagnostycznych służy zintegrowany w tej platformie mechanizm zarządzania informacją [54].

Z taką myślą powstał prototyp Sieciowego Monitora Obiektu, aplikacji umożliwiającej nie tylko ciągłą kontrolę parametrów obiektów przemysłowych i biznesowych, ale również zarządzanie grupami pracowników oraz nowoczesnym przepływem informacji (technicznej i korporacyjnej). Opracowane rozwiązanie stanowi obiecujący początek dla wszechstronnego rozwoju takich aplikacji. Dlatego zamierzeniem dalszych kroków jest poszerzenie funkcjonalności i poprawa właściwości technicznych SMO poprzez dostosowanie systemu do praktycznych wymagań stawianych przez nowoczesne firmy [57].

4

JĘZYK SMOL JAKO NARZĘDZIE OPISU SIECI POMIAROWO-WYKONAWCZYCH

Rozdział stanowi studium ukazujące możliwości wykorzystania koncepcji języków DSL we współczesnej automatyce i informatyce stosowanej. W rozdziale zaprezentowane są założenia koncepcyjne języków domenowych oraz możliwości ich wykorzystywania w rozwiązywaniu określonych grup problemów. Zaprezentowana jest szczegółowa analiza dotycząca samego języka SMOL. Opisane są metody oraz mechanizmy opracowane dla języka SMOL (rodzaje węzłów, funkcje transformujące itd.). Uzasadniany jest sposób wyboru technologii (język programowania) oraz implementacji. Szczegółowo omawiane są mechanizmy testowania i weryfikacji systemu SMO. Rozdział zawiera też informacje na temat technicznych możliwości rozwoju projektu poprzez tworzenie nowych modułów funkcjonalnych.

4.1. Wprowadzenie

Ekonomiczne kwestie sprawiają, że szukamy nowych możliwości lepszego rozwiązywania konkretnych problemów technicznych. Dzięki wiedzy, technologii i doświadczeniu, można osiągnąć większą niezawodność i skuteczność (optymalności) projektowanych systemów.

Współczesne systemy automatyki oparte są w znacznej mierze na rozbudowanych sieciach pomiarowych. Aktualne pomiary dają nam możliwość szybkiego zweryfikowania poprawności działania systemu. Dynamiczny rozwój technologii sprawia, że komputerowe systemy automatyki mają coraz większe możliwości analizy i wykorzystania danych pomiarowych (często zapisywanych w centralnym miejscu). Współcześnie stosowane techniki analizy danych (ang. *data mining*) pozwalają na opracowanie użytecznych

praktycznie mechanizmów detekcji lub predykcji uszkodzeń, opierając się na symptomach ukrytych w danych pomiarowych. Awaria dużego systemu zarządzania bądź automatyki zwykle oznacza całkowite lub częściowe zatrzymanie procesu oraz straty finansowe. Dlatego narzędzia pozwalające na modelowanie i symulację takich systemów są niezwykle istotne dla analizy, która ma na celu identyfikację słabych punktów budowanego rozwiązania sterującego.

W ramach prowadzonych prac badawczych opracowano system [55, 59, 61] pozwalający na opis struktury sieci połączonych ze sobą urządzeń w dedykowanym języku domenowym SMOL[1] oraz przeprowadzenie odpowiedniej symulacji.

W niniejszej rozdziale przedstawiona zostanie koncepcja języka, semantyka oraz praktyczny przykład ilustrujący możliwości opracowanego środowiska.

4.2. Założenia języka SMOL

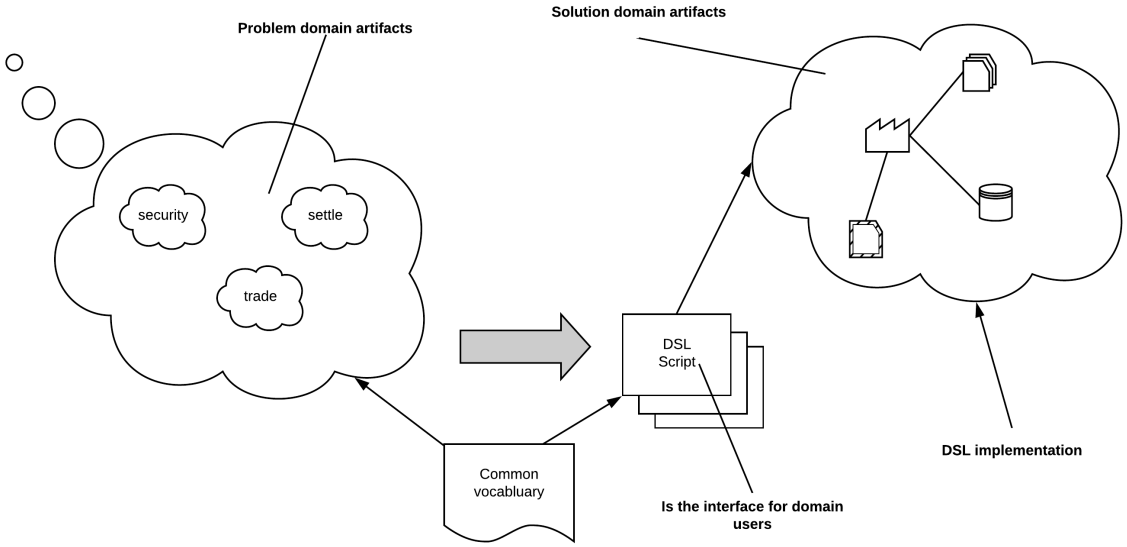
Rozwój techniki pomiarowej spowodował wykształcenie się gałęzi rozproszonych systemów pomiarowych, które są obecnie bardzo często wykorzystywane do inteligentnego zarządzania i sterowania obiektami. Dynamiczny rozwój tej gałęzi nauki stanowi też ważny czynnik inspirujący autora do opracowania prezentowanego mechanizmu modelowania struktur sieci PDS[2] (pomiarowo-diagnostyczno-sterujących). W niniejszej pracy skupiamy się na propozycji metody opisu rozbudowanych sieci PDS, opartej na dedykowanym języku SMOL. Opis struktury realizowanej sieci za pomocą języka SMOL stanowi podstawę do późniejszej analizy sieci oraz jej optymalizacji.

4.3. Teoria języków domenowych

Grupy języków domenowych (DSL), w odróżnieniu od języków programowania ogólnego przeznaczenia (GPLs[3]), wyróżniają wyeksponowane charakterystyczne cechy pozwalające na stosunkowo prosty opis danej dziedziny. Umożliwia to tworzenie unikatowych narzędzi dopasowanych do rozwiązywania konkretnych grup problemów. W wielu przypadkach budowa rozwiązań domenowych ma swoje uzasadnienie i przekłada się bezpośrednio na szybkość, jakość oraz bezawaryjność budowanych aplikacji. Niestety tworzone w ten sposób języki mają również swoje wady chociażby w kontekście użytkowania (nowa składnia, formuła) i dodawania nowych funkcjonalności. Ich unikatowość oraz ściśle określone miejsce zastosowania sprawiają, że stają się rozwiązaniami niszowymi, tym samym zmniejszając swoje możliwości na równie dynamiczny rozwój, co języki z rodziny GPL.

[1] ang. *Language for Networked Systems for Monitoring Objects.*
[2] komputerowa sieć Pomiarowo-Diagnostyczno-Sterująca.
[3] ang. *General Purpose Languages.*

Zastanawiając się nad stworzeniem rozwiązania bazującego na języku DSL warto również wziąć pod uwagę przyszłość rozwijanej koncepcji. W przypadku narzędzi programowych wiąże się ona w znacznej mierze z możliwościami utrzymania i rozwoju stworzonej aplikacji. Utrzymanie języków domenowych może to okazać się dość problematyczne ze względu na ograniczoną liczbę specjalistów oraz zakres wiedzy eksperckiej wymaganej do opisu danej dziedziny. Grupy języków GPL służących do rozwiązywania dowolnej grupy problemów mogą pod pewnymi względami wydawać się lepsze, niestety ich uniwersalność i wszechstronność może okazać się również swojego rodzaju wadą.



Rys. 4.1. Wykorzystanie języka domenowego do opisu problemu z danej dziedziny [27].

Szczególnie w przypadku rozwiązywania określonych grup problemów, sposób myślenia człowieka jest na wyższym poziomie abstrakcji niż opis stosowany w języku maszynowym. Bazując na definicjach bezpośrednio związanych z danym procesem w stosunkowo łatwy sposób można opisać go w formie programu. Właśnie w takich przypadkach języki domenowe pokazują swoją prawdziwą siłę. Opis problemu z danej dziedziny staje się prosty i zrozumiały dla ekspertów pracujących nad jego rozwiązaniem a specyficzne nazewnictwo - częścią programu.

Zgodnie z założeniami koncepcji budowy języków DSL, powinny one pozwalać na opisywanie pewnych klas problemów za pomocą charakterystycznej terminologii domenowej. Sposób algorytmizacji problemu z wykorzystaniem języka domenowego przedstawiony została na rys. 4.1 [27].

W DSL wychodzi się z założenia, że użytkownik języka wcale nie musi być programistą a jedynie posiadać ekspercką wiedzę ze swojego zakresu. Język powinien być na tyle pro-

sty, aby umożliwić algorytmizację problemu za pomocą terminologii charakterystycznej dla określonej dziedziny. Istotne jest również, aby zapewniał mechanizm bezpośredniego mapowania pomiędzy terminologią charakterystyczną dla danego zagadnienia a abstrakcyjną reprezentacją w formie programistycznego obiektu. Struktura i terminologia języka DSL musi być na tyle intuicyjna, że jej zrozumienie nie powinno stanowić większego problemu dla użytkownika nie będącego programistą [27].

4.3.1. Dychotomie w językach DSL

Z założenia języki DSL należą do grupy rozwiązań dychotomicznych. Dychotomia w tym przypadku związana jest z ich dwojakim postrzeganiem przez pryzmat charakterystyki i definicji samego języka. Oczywiste jest odwoływanie się do grupy języków ogólnego przeznaczenia (GPL) w przypadku dokonywania porównania cech poniżej.

Jedną z płaszczyzn porównania jest odpowiedz na pytanie, czy języki DSL zaliczają się do grupy **abstrakcyjnych** (ang. *abstract*), czy bardziej **konkretnych** (ang. *concrete*). Z założenia im język programowania jest bardziej abstrakcyjny tym bardziej ukrywa przez programistą semantykę operacyjną [75]. Wykorzystywany jest tu mechanizm enkapsulacji wiedzy symantycznej, przechowywanej w formie **niejawnej** (ang. *implicitly*) – odwrotnie niż w przypadku języków GPL, gdzie do czynienie mamy z formą **wyraźną** (ang. *explicitly*). Składania taka pozwala na uproszczenie semantyki samego języka domenowego i wyeksponowanie jedynie elementów najbardziej istotnych.

Kolejną wartą rozważenia dychotomią jest podział na języki **niskopoziomowe** i **wysokopoziomowe**. Ta płaszczyzna podziału jest w pewien sposób pochodną powyższych rozważań. Języki DSL są wysokopoziomową formą reprezentacji logiki. Wysoka forma abstrakcji pozwala skoncentrować się na rozwiązaniu problemu, a nie na poszukiwaniu metod implementacji. Taki sposób tworzenia implementacji jest szczególnie istotny dla osób nie do końca wyspecjalizowanych w programowaniu, a posiadających ekspercką wiedzę z zakresu danego procesu/zagadnienia opisywanego językiem DSL.

4.3.2. Koncepcja języka SMOL

SMOL – jako język dedykowany o semantyce pozwalającej na modelowanie komputerowych sieci pomiarowo-diagnostyczno-sterujących, stanowi przykład narzędzia dziedzinowego, tj. służącego do rozwiązywania problemów z określonego obszaru [11].

Opis struktury sieciowej stanowi niezbędną podstawę reprezentacji problemów konfiguracyjnych, restrukturalizacji, bądź optymalizacji podukładów takiej sieci. Język SMOL służy zatem do opisu elementów składowych sieci diagnostycznej i towarzyszących im połączeń wraz z odpowiednią parametryzacją. Zastosowany formalny zapis pozwala też na weryfikację perspektywicznego procesu realizacji takiej sieci przy użyciu stosownego par-

sera, który stanowi programistyczną platformę analizy zdefiniowanej struktury sieciowej oraz jej parametrów.

Semantyka języka SMOL opiera się na strukturach przetwarzania hierarchicznego. Zakłada się przy tym, że wszystkie sygnały sieciowe koncentrują się ostatecznie w węźle głównym (ang. *Central Node*), który jest centralnym wierzchołkiem grafu.

4.3.3. Elementy opisu sieci diagnostyczno-pomiarowych

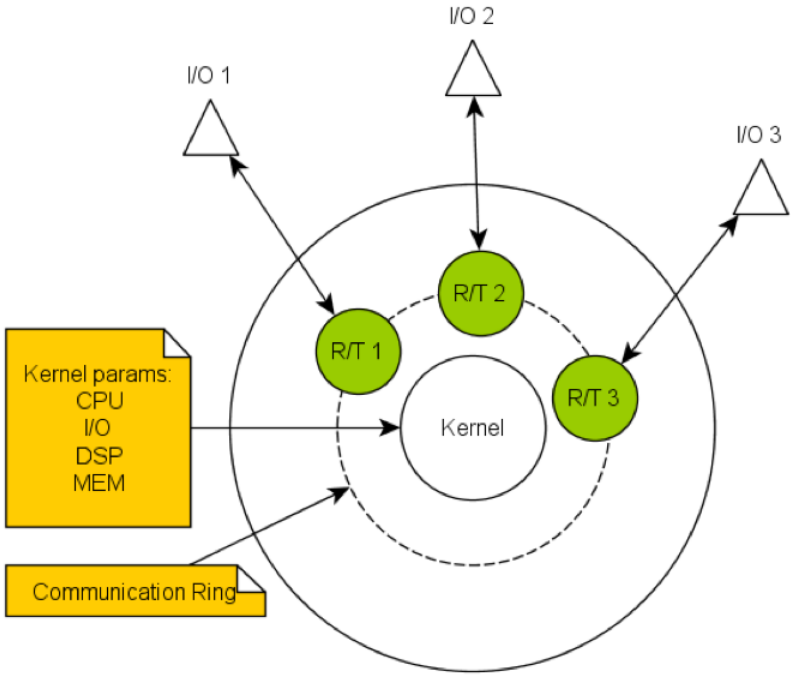
Jednym z najlepszych sposobów modelowania przemysłowych sieci diagnostycznych jest wykorzystanie skierowanego grafu przepływowego jako elementu reprezentacji relacji występujących w sieci połączeń. Model taki reprezentuje trójka zbiorów (zbiór wierzchołków, zbiór krawędzi grafu, oraz zbiór odwzorowań). Zdefiniowanie tych zbiorów pozwala na wyrażenie rzeczywistych przepływów informacji w sieciach diagnostyczno-pomiarowych. Poniżej przedstawiamy podstawowe typy węzłów wykorzystywane w takim grafie do prawidłowego odwzorowania relacji sieciowych. Główny podział dotyczy stosowania węzłów statycznych i dynamicznych, które implementują funkcje transformujące, opisane w p.(4.3.7).

4.3.4. Węzeł centralny CN (*Central Node*)

Węzeł centralny (element CN), sieci jest w ogólności obiektem typu MIMO[4] (z wieloma wejściami i wyjściami). Jest elementem niezbędnym do zdefiniowania praktycznej sieci PDS. W rzeczywistej komputerowej sieci połączeń element ten stanowi węzeł centralny dla przesyłanych informacji, do którego spływają dane pochodzące z urządzeń pomiarowych oraz z którego mogą wychodzić sygnały sterujące. Schemat takiego węzła przedstawiono na rys. 4.2.

Węzeł centralny (rys. 4.2) posiada grupę kanałów komunikacyjnych (CC[5]) oraz przypisane im moduły R/T[6], należące do warstwy komunikacyjnej nazywanej też pierścieniem komunikacji (ang. Communication Ring) i służące do wymiany informacji pomiędzy jądrem (ang. Kernel) a innymi węzłami. Jądro węzła CN implementowane jest jako system komputerowy, definiowany poprzez swoje zasoby (i parametry) sprzętowe (CPU[7] , I/O[8], DSP [9] oraz MEM[10]).

[4]ang. *Multiple Input Muliple Output.*
[5]ang. *Communication Channel.*
[6]ang. *Receiver/Transmiter.*
[7]ang. *Central Processing Unit.*
[8]ang. *Input/Output.*
[9]ang. *Digital Signal Processing.*
[10]ang. *Memory.*



Rys. 4.2. Central Node: Węzeł centralny, reprezentujący obiekt MIMO (wielowymiarowy).

Węzeł centralny jest elementem wymaganym, jeżeli zamierzamy weryfikować realizowalność projektowanej sieci.

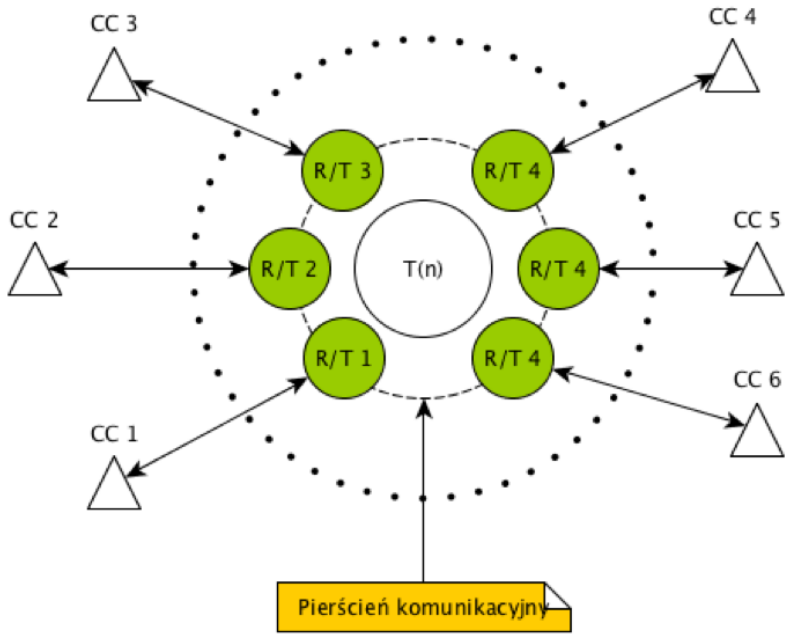
4.3.5. Węzeł przenoszący TN (*Transfering Node*)

Węzeł przenoszący (przekaźnik, element TN) to uniwersalny węzeł, w ogólności system typu MIMO, który w sieci pełni funkcję elementu przekaźnikowego o strukturze pierścieniowej przedstawionej na rys. 4.3), gdzie elementy CC oraz R/T spełniają funkcje analogiczne do opisanego wcześniej węzła centralnego CN.

Jądro węzła może posiadać zaimplementowane funkcje transformujące $T_n(n)$. Koncepcja wykorzystania *transformatorów* przedstawiona została w punkcie 4.3.7.

Elementom przekaźnikowym TN w rzeczywistych realizacjach sieci PDS przypisać można określone wejścia i wyjścia oraz sposób transmisji sygnału. W praktyce stosuje się transmitery pracujące w standardzie przewodowym lub bezprzewodowym (WiFi[11], ZigBee[12], Switch[13], Modem).

[11] Wireless Fidelity.
[12] IEEE 802.15 standard.
[13] http://en.wikipedia.org/wiki/Network_switch.



Rys. 4.3. Transfering Node: Uniwersalny węzeł typu MIMO.

4.3.6. Expander EX

W rozwiązaniach praktycznych powszechnie stosuje się szynę danych jako wyróżniony element wykorzystywany do transmisji danych (pomiarowych i sterowania). Taką funkcję powinien zatem uwzględniać proponowany język SMOL jako narzędzie projektowania przemysłowych sieci *pomiarowo-diagnostyczno-sterujących*.

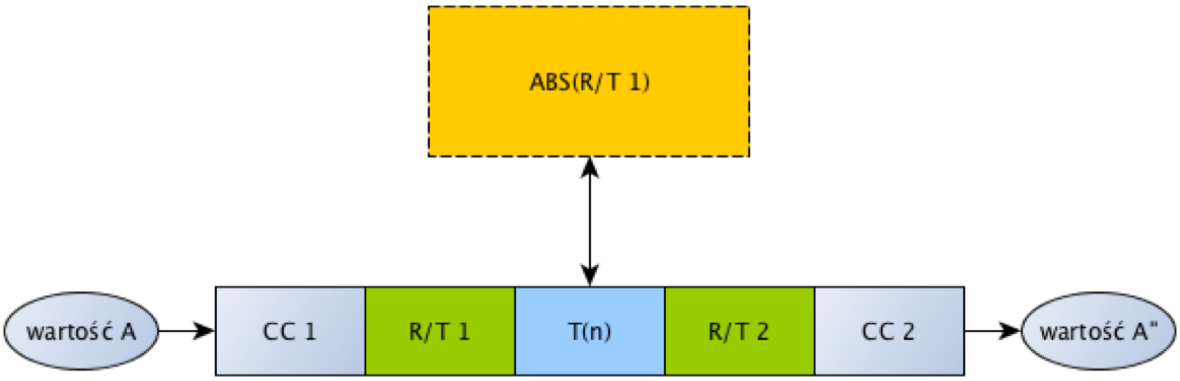
Expander, reprezentujący transmisyjny element sieci, pozwala na wymianę informacji w oparciu o powszechnie stosowane w automatyce i systemach pomiarowych magistrale (PROFIBUS[14], CAN[15], RS-232, RS-485, DSB[16] oraz KNX/EIB[17]). Typ wykorzystywanego protokołu definiuje się w liście parametrów dostępnych dla tego obiektu.

4.3.7. Transformator TR – funkcja transformująca

W systemach PDS często wymaga się szybkiej i raczej prostej modyfikacji strumienia przesyłanych danych. Transformatory (TR) pozwalają zatem na realizację zbioru uniwersalnych funkcji matematycznych (suma, różnica, iloczyn, czy iloraz, jak również: różniczkowanie, całkowanie, uśrednianie, albo bardziej złożone filtry cyfrowe), operujących

[14]https://en.wikipedia.org/wiki/Profibus.
[15]https://en.wikipedia.org/wiki/CAN_bus.
[16]Diagnostic Service Bus.
[17]https://en.wikipedia.org/wiki/KNX_(standard)

na danych wejściowych. Aktualnie możliwość zastosowania funkcji transformującej przewidziana jest jedynie dla grupy węzłów CN (4.3.4) i TN (4.3.5).



Rys. 4.4. Węzeł TN z zaimplementowaną jako transformator TR jako funkcją transformującą T(n).

Mechanizm działania transformatorów sprowadza się do operacji na danych trafiających do węzła poprzez port komunikacyjny wejściowy (CC_1) i przekazywanych na port komunikacyjny wyjściowy (CC_2). Na rys. 4.4 przedstawiono schemat transformatora modyfikującego dane trafiające do portu komunikacyjnego CC_1 przy użyciu funkcji transformującej $T(n)$, realizującej przykładową funkcję $abs(x) = |x|$, oraz 'wystawianych' na porcie komunikacyjnym CC_2.

4.3.8. Parametry połączeń pomiędzy węzłami

Definiując parametry połączeń pomiędzy węzłami (czyli parametry kanałów komunikacyjnych CC) precyzuje się charakter wykorzystywanego medium do transportu danych (np. sieć typu Ethernet). Projektant uzyskuje tu możliwość określenia fizycznych parametrów kanału komunikacyjnego, tj. długości fizycznego medium, rodzaju medium (np. powietrze, miedziany przewód, sieć komputerowa), tłumienności, oraz szacowanej maksymalnej prędkości przesyłu danych.

4.3.9. Sensor/Actuator Node (*SAN*)

SAN stanowi typ węzła kończącego strukturę sieci w danej gałęzi. W rzeczywistych sieciach diagnostycznych element ten obsługuje (opisuje) układy pomiarowe lub stany logiczne układów sterowania.

4.4. Środowisko języka SMOL

Koncepcja języka SMOL oparta została na rozwiązaniu spełniającym wymagania paradygmatu programowania obiektowego (abstrakcja, polimorfizm, hermetyzacja oraz dziedziczenie) [59]. Taka konstrukcja ułatwia odwzorowanie relacji występujących w rzeczywistych systemach.

Zapisany w języku SMOL program wyraża zbiór komunikujących się między sobą procesów (obiektów) w celu realizacji określonych zadań. Zaprezentowane rozmaite autorski typy węzłów [58] pozwalają na implementację obiektów odpowiadających swoimi funkcjami urządzeniom wykorzystywanym powszechnie w komunikacji, diagnostyce i automatyce. Obiekty takie – wyrażone zgodnie z koncepcją języków programowania wysokiego poziomu – posiadają strukturę zdefiniowaną za pomocą zmiennych pól oraz zachowań – metod czyli funkcji wbudowanych w obiekt (funkcja występująca w kodzie samodzielnie – to zwykła funkcja). Z punktu widzenia programisty, pozwala to jednoznacznie zdefiniować role pełnione przez obiekty w programie oraz porządkować strukturę kodu. Opierając się na języku SMOL, projektant podczas modelowania struktury sieci nie musi ograniczać się do wykorzystania dostarczonych gotowych implementacji obiektów. Dzięki zastosowaniu otwartej architektury możliwe jest dopisanie (z wykorzystaniem mechanizmów dziedziczenia oraz odpowiednich interfejsów) własnych zachowań projektowanych elementów systemu PDS. Umożliwia to łatwą rozbudowę środowiska SMOL a tym samym zwiększa jego uniwersalność.

W skład środowiska SMOL wchodzi również pakiet gotowych obiektów i funkcji. Dla ułatwienia pracy projektanta, obiekty podzielono na dwie główne grupy.

SMOH to grupa obiektów odzwierciedlających elementy sprzętowe. Pozwala ona użytkownikowi na skorzystanie z gotowych implementacji obiektów, które reprezentują najczęściej używane urządzenia w rozważanych sieciach. Przykładem takiej gotowej implementacji jest przełącznik sieciowy (ang. *switch*), czujnik pomiarowy, czy aktuator (układ wykonawczy). Każdy z gotowych elementów posiada mechanizmy umożliwiające połączenie go z innymi elementami sieci. Obiekty łączą się ze sobą poprzez ściśle zdefiniowane interfejsy (obiekty nie posiadające określonego interfejsu nie pozwalają na połączenie z innym obiektem wymagającym interfejsu do komunikacji). Opierając się na dostarczonym podstawowym zbiorze elementów i funkcji, projektant może budować własne typy obiektów z wykorzystaniem mechanizmu dziedziczenia.

SMOF stanowi zbiór zaawansowanych funkcji użytecznych w implementacji procesu przepływu i przetwarzania danych w sieciach PDS. W skład pakietu systemowego wchodzą funkcje transformujące. Pakiet SMOF zawiera m.in. zaimplementowany algorytm Dikstry, umożliwiający znalezienie najkrótszej drogi łączącej wykorzystywane w sieci węzły oraz algorytm Edmondsa Karpa, pozwalający określić maksymalną przepływność w całej sieci lub w określonym fragmencie. Pakiet SMOF zapewnia również możliwość wy-

generowania graficznej reprezentacji modelowanego środowiska. Zobrazowanie struktury połączeń pozwala łatwiej pojąć sieć i zależności pomiędzy jej poszczególnymi elementami, ma wpływ na ostateczną ocenę użyteczności sieci oraz ułatwia optymalizację sieci.

4.4.1. Koncepcja budowy parsera języka SMOL

Parser SMOL to narzędzie służące do 'wirtualnej implementacji' programu zapisanego w języku SMOL. Proces przetwarzania kodu przez parser rozpoczyna się od analizy leksykalnej, w której sprawdzana jest poprawność użytych elementów i funkcji. Parser zwraca informację, w której przedstawia wykryte błędy oraz miejsce ich występowania w analizowanym kodzie. Kolejnym etapem jest wykonanie zapisanych w kodzie programu funkcji.

Do zadań języka SMOL należy wygenerowanie modelu w formie ekstraktu wczytywanego przez symulator. Model taki zawiera definicję wszystkich połączeń zapisaną w formie pliku GaphML[18] oraz informacje o wykorzystywanych w sieci funkcjach transformujących.

4.5. Opis architektury opracowanego środowiska

Budowane rozwiązanie oparte zostało na technologii Groovy. Jest to obiektowy język skryptowy wzorowany na składni Javy i uruchamiany na wirtualnej maszynie JVM[19]. Środowisko JVM pozwala wykonywać program skompilowany do postaci kodu bajtowego, zapewniając tym samym mechanizmy do zarządzania pamięcią i natywną obsługą wyjątków. Groovy dzięki wykorzystaniu dynamicznego typowania, domknięć, przeciążeniu operatorów oraz wsparcia dla meta-programowania (meta-klasy, kategorie, transformacje AST[20]), doskonale nadaje się do budowania własnego języka domenowego. Wykorzystana technologia pozwala na dogodną integrację języków SMOL i Groovy. Istnieje ponadto możliwość przeplatania języków między sobą w celu osiągnięcia większej uniwersalności budowanego rozwiązania. Projektant systemu posiada zatem możliwość implementacji części realizowanej funkcjonalności w środowiska Java, a następnie wywołaniu jej poprzez język Groovy, bezpośrednio w programie języka SMOL. Takie rozwiązanie daje wielkie możliwości rozbudowy realizowanych modeli sieci PDS.

4.5.1. Semantyka języka SMOL

Poniżej przedstawiono fragment listingu SMOL opisującego sieć PDS, wykorzystywaną w inteligentnym budynku. Pełny kod programu w języku SMOL opisujący strukturę

[18] http://en.wikipedia.org/wiki/GraphML.
[19] ang. *Java Virtual Machine.*
[20] https://sewiki.iai.uni-bonn.de/research/jtransformer/start

zaprezentowanej sieci PDS umieszczony został w dodatku B.

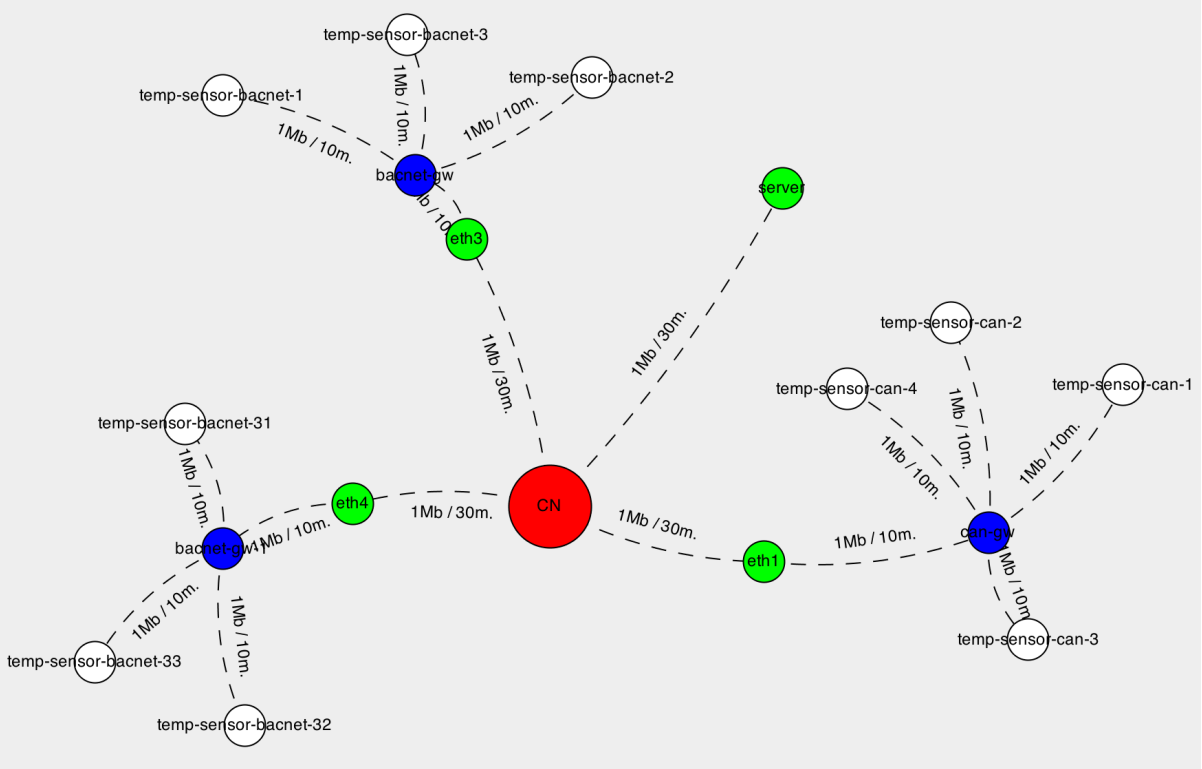
```
/*
 * Transfer node definition
 * Define a network adapter
 * Assign IP address
 */
tn "eth1", {
    ip "1"
}

/*
 * Define a network adapter
 * Assign IP address and name
 */
tn "Server", {
    ip "2"
}

/*
 * Expander node definition
 * Define CAN network gateway
 * Connect to the Transfer node
 */
expander "can-gw", {
    connect "eth1"
    model "bacnet"
}

/*
 * Expander node definition
 * Define Bacnet network gateway
 * Connect to the Transfer node
 */
expander "bacnet-gw", {
    connect "eth3"
    model "bacnet"
}
```

```
/*
 * Temp sensor definition
 * Connect sensor to CAN bus
 * Set measure values final destination address (2)
 * Set a measure generation frequency
 */
san "temp-sensor-can-1",{
    connect "can-gw"
    destAddress "2"
    freq "2000"
}

/*
 * Run map function which draws a graph with all connections
 */
draw "map", {
    fullmap "true"
}

/*
 * Run simulation phase
 */
sim "run", {
    stop "10s"
}
```

Na rys.4.5 przedstawiono schemat wygenerowanej graficznej reprezentacji sieci połączeń na podstawie kodu programu napisanego w języku SMOL.

W skład zaprezentowanej sieci wchodzi zespół komunikujących się ze sobą temperaturowych czujników pomiarowych oznaczonych na rysunku symbolem temp-sensor (jako realizacja węzła typu SAN; patrz pkt. 4.3.9). Czujniki przekazują informację pomiarową z wykorzystaniem magistrali CAN oraz Bacnet (p. 4.3.9) do systemu komputerowego CN (p. 4.3.4). Na schemacie wyróżniony został też element przenoszący TN (p. 4.3.5) realizujący funkcję MISO[21] na przesyłanych danych. Na schemacie przedstawiono również wybrane przez projektanta (z grupy dostępnych opcji opisanych w p. 4.3.8 parametry połączeń (maksymalna przepustowość kanału komunikacyjnego) pomiędzy węzłami sieci.

[21] ang. *Multiple Input Single Output.*



Rys. 4.5. Schemat wygenerowanej sieci połączeń po uruchomieniu parsera SMOL.

4.6. Podsumowanie

Możliwość szybkiej wizualizacji budowanego rozwiązania PDS jest jednym z celów prezentowanej platformy. Projektant sieci otrzymuje funkcjonalne narzędzie pozwalające opisać strukturę oraz związane z monitorowanym obiektem parametry i relacje.

Modelowanie struktur sieci diagnostyczno-pomiarowo-sterujących w języku SMOL z wykorzystaniem uniwersalnego i otwartego środowiska pozwala projektantowi na korzystanie z systemowych (autorskich) gotowych implementacji elementów sieciowych oraz na tworzenie i późniejsze wykorzystywanie własnych modeli elementów w realizowanych projektach.

Zaprezentowana koncepcja odwzorowywania struktury rzeczywistych układów sieciowych pozwala w prosty sposób opisać i zwizualizować modelowaną sieć. Załączony przykład ilustruje koncepcję opracowanej semantyki polegającej na opisie: (1) poszczególnych elementów rozważanej sieci diagnostycznej, która może być wykorzystana w monitorowanym obiekcie przemysłowym, oraz (2) mechanizmów (relacji) występujących pomiędzy tymi elementami.

Możliwość swobodnego definiowania struktur sieci PDS z wykorzystaniem języka SMOL stanowi wstępny etap do realizowanego większego projektu, związanego z budową

zintegrowanego narzędzia do symulacji i optymalizacji sieci. W kolejnym etapie rozwoju tego projektu zaprezentowany zostanie stowarzyszony z językiem SMOL symulator sieci pomiarowo-diagnostyczno-sterujących.

5

SYSTEM SYMULACJI OPARTY NA JĘZYKU SMOL

Wstęp do rozdziału stanowi studium symulacji dyskretnej. W dalszej części rozdziału scharakteryzuje się sposoby i techniki symulacji (procesowej, zdarzeniowej oraz mieszanej), jak też ich wykorzystanie w środowisku SMOLSim. W rozdziale omawia się zagadnienie wyboru odpowiednich ram programowych (ang. *framework*) wspomagających realizację symulacji, w tym problem synchronizacji pomiędzy zdarzeniami na wspólnej osi czasu. Dokonuje się przeglądu darmowych narzędzi opracowanych dla platformy Java (DESMO-J) oraz Python (SimPy). Przedstawia się opis zastosowania języka SMOL oraz opracowanego parsera do generowania kodu symulacyjnego. Podaje się też przykład ilustrujący metodę opisu sieci PDS w języku SMOL oraz sposób przeprowadzania symulacji w środowisku SMOLSim.

5.1. Wprowadzenie

Potrzeba stworzenia warunków zbliżonych do rzeczywistych, które pozwalają na weryfikację działania układów sterownia i diagnostyki, stanowiła główną motywację do opracowania środowiska symulacyjnego SMOLSim. W wyniku prowadzonych badań opracowano rozwiązanie w postaci narzędzia, które pozwala na weryfikację jakości działania zaprojektowanej konfiguracji sprzętowej w ściśle określonych warunkach. Do zadań symulatora należy zarówno generowanie warunków brzegowych i początkowych symulowanej sieci PDS, jak i odpowiednia reakcja sieci na wewnętrzne sygnały związane ze współpracą między urządzeniami.

W ramach prac projektowych podjęto się opracowania systemu symulacji dyskretnej,

pozwalającego na modelowanie i symulację określonych typów sieci. Wynikiem przeprowadzonych prac projektowych i implementacyjnych jest narzędzie pozwalające przeprowadzić weryfikację realizowalności i efektywności budowanego rozwiązania pomiarowo-diagnostycznego-sterującego.

Niniejszy rozdział stanowi autorskie studium przedmiotowego modułu systemu symulacji SMOLSim opartego na języku SMOL.

5.2. Systemy symulacyjne

Jednym z założeń dotyczących wykorzystania symulacji jest możliwość modelowania rzeczywistych systemów oraz weryfikacji realizowalności wymyślonych struktur lub konfiguracji. Forma wirtualnej obserwacji pozwala zdecydowanie lepiej zrozumieć symulowany proces oraz jego reakcję na różne pobudzenia. Takie podejście wymaga dogłębnej wiedzy na temat projektowanego systemu (modelu sieci), jego wymagań oraz zasad optymalizacji, jak również specyfiki symulacyjnego środowiska.

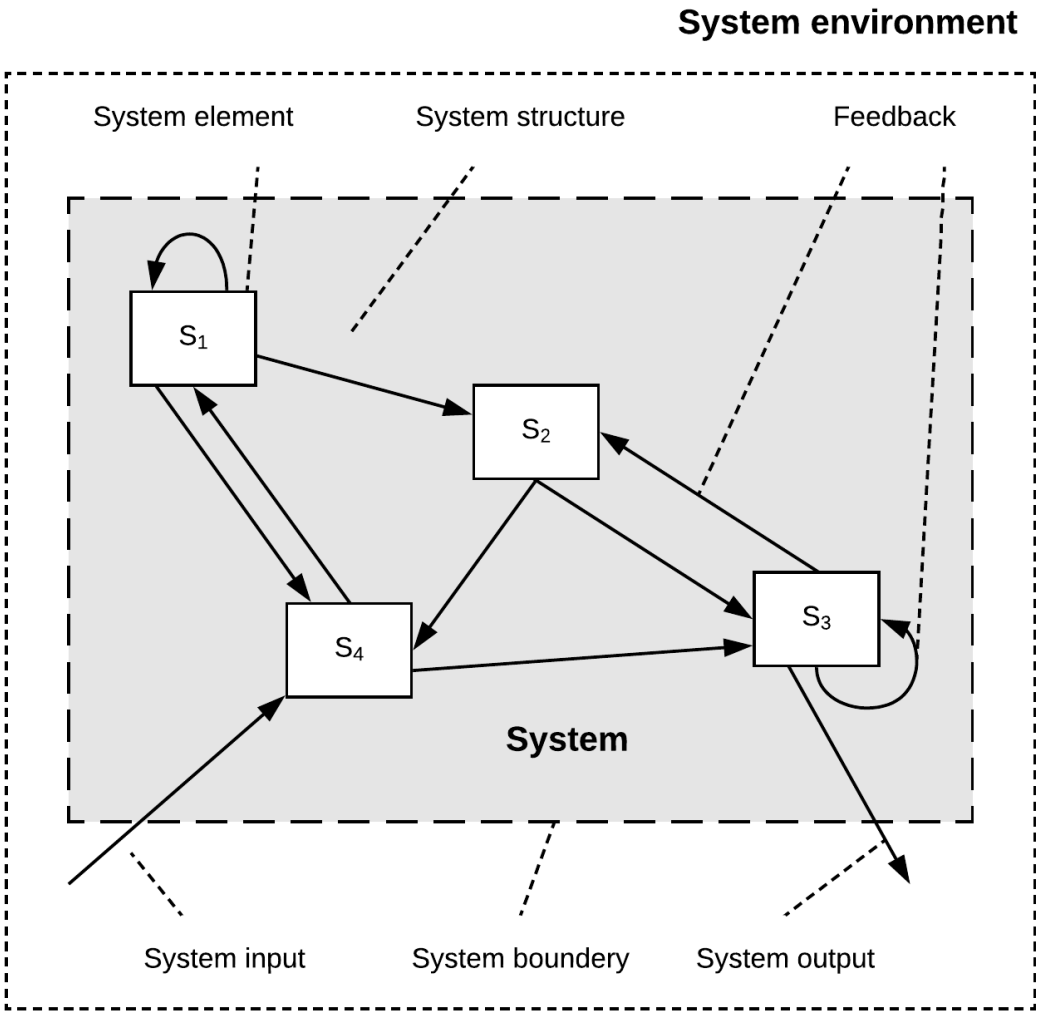
Według Page'a i Kreutzer'a [84] opracowywanie modeli symulacyjnych opiera się na nauce określanej jako badania operacyjne (ang. *operations research*), która pozwala na wykorzystywanie metodologii taktycznego i strategicznego podejmowania decyzji oraz planowania. W ramach tej nauki wykorzystuje się odpowiednio skwantowaną informację w celu znalezienia rozwiązania optymalnego dla modelu analitycznego lub prowadzi analizę w przestrzeni stanów w oparciu o model symulacyjny.

Poniżej przedstawiamy podstawowe definicje wykorzystywane w tym procesie. Taka forma podziału pozwala na ustandaryzowanie terminologii używanej w dalszej części rozdziału.

System określany jako dobrze opisany (z operacyjnego punktu widzenia) fragment świata rzeczywistego lub wirtualnego. Zawiera on jednoznacznie identyfikowalne komponenty oraz zmienne pozwalające na opis zachodzących interakcji. Rzeczywiste zachowania komponentów mogą być nieintuicyjne i jednocześnie trudne do przewidzenia. Przy analizie działania systemów inżynierskich i przemysłowych wykorzystuje się techniki oparte na abstrakcji i agregacji. Wymagana jest wówczas odpowiednia wiedza o wszystkich komponentach systemu, ich właściwościach, akcjach, jak również interakcjach pomiędzy poszczególnymi elementami. Jedną z form analizy systemów jest *analiza liniowa*(*strukturalna*), polegająca na podziale złożonego systemu na mniej złożone wyizolowane elementy. Oczywiście trudniejsze w analizie są systemy współbieżne, z dynamicznie zmieniającymi się komponentami. Modelowanie wymaga wówczas złożonej analizy zachodzących zjawisk (nieliniowych, wariantnych lub niestacjonarnych).

Zgodnie z teorią systemów [84], analiza polega na wyszukiwaniu pewnych wzorców oddziaływania między komponentami systemu. Natomiast ogólne zasady dotyczące dekom-

pozycji determinują strukturę modelowanego systemu oraz możliwości opisu wszystkich jego funkcji. Na rys. 5.1 przedstawiono przykładową strukturę systemu.



Rys. 5.1. Przykładowa struktura systemu oraz środowiska [84].

Przy modelowaniu systemów można wyróżnić następujące elementy składowe:

- komponenty - podstawowe elementy systemy (często niepodzielne, atomowe)
- właściwości lub atrybuty – zmienne odwzorowujące aktualny stan systemu (w tym poszczególnych komponentów systemu)

- stan systemu – wartości przechowywane w wektorze zmiennych stanu w określonej chwili czasu
- zachowanie systemu – trajektoria wektora stanu systemu w określonym przedziale czasu
- złożoność systemu – utożsamiana z liczbą zmiennych stanu lub liczbą połączeń pomiędzy komponentami
- granica systemu – linia oddzielająca system od środowiska
- system otwarty – wchodzący przynajmniej w jedną interakcję ze środowiskiem
- system zamknięty – system niekomunikujący się ze środowiskiem
- system statyczny – reagujący jedynie na sygnały wejściowe (nie odnoszący się w swojej reakcji ani do czasu, ani do przeszłości)
- system dynamiczny – system uzależniony nie tylko od wejścia (posiadający jakąś formę pamięci wewnętrznej).

Model jest niezbędnym elementem systemu symulacji. Głównym jego zadaniem jest definiowanie zachowania, czyli interakcji pomiędzy poszczególnymi komponentami opisywanego systemu. Modelowanie opiera się na empirycznej obserwacji połączonej z umiejętnością stawiania hipotez, na podstawie których powstają logiczne konkluzje (implikacje) pozwalające na zalgorytmizowany opis działania systemu (lub jego fragmentu). Warto zaznaczyć, że jakość budowanego modelu zależna jest od wielu czynników. Jednym z głównych jest szeroko pojęta wiedza samego twórcy modelu oraz doświadczenie na temat modelowanego procesu – im bliżej wiedzy eksperckiej, tym większa szansa na sukces oraz dokładniejsza forma odwzorowania. Ważne podczas projektowania i budowy modelu są również stawiane cele, które pozwalają na uzyskiwanie właściwego modelu oraz ułatwiają zrozumienie systemu. Ponadto poprawnie przeprowadzony proces modelowania umożliwia dokładną predykcję albo opracowanie skuteczniejszych mechanizmów monitoringu lub sterowania.

Jedną z form opisu układu rzeczywistego w postaci implementowanego modelu jest mapowanie interakcji pomiędzy poszczególnymi jego elementami z wykorzystaniem abstrakcji i idealizacji. Zastosowanie tej metodologii pozwala na analizę nawet bardzo złożonych systemów w dogodnej formie, pozwalającej na ich późniejszą modyfikację i rozwój. W przypadku symulacji określonego fragmentu systemu mamy możliwość opracowania różnych alternatywnych modeli, pozwalających na analizę działania lub weryfikację skuteczności doboru parametrów w zmiennych warunkach.

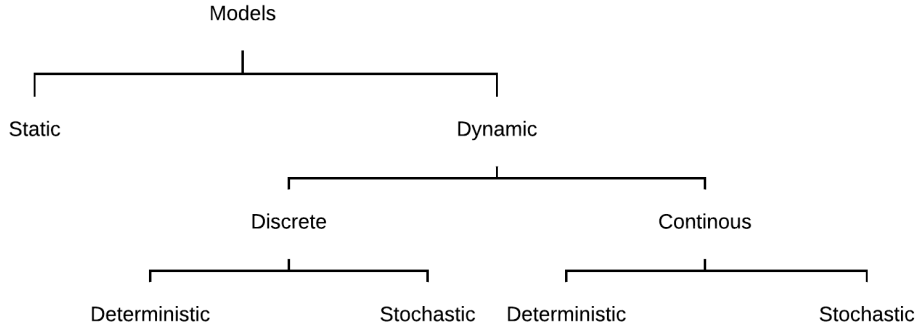
Modele mogą podlegać też kategoryzacji w zależności od sposobu ich reprezentacji

lub metody analizy.

Jedną z podstawowych form kategoryzacji jest podział modeli na:

- fizyczne (np. skalowalny model statku)
- werbalne (np. instrukcja określająca opis drogi do przebycia)
- opisowe graficzne (np. diagram w języku UML)
- graficzne (np. sieci Petriego)
- abstrakcyjne, oparte na równaniach matematycznych (np. równania różniczkowe)
- abstrakcyjne - algorytmiczne (np. model dyskretnej symulacji zdarzeniowej).

Na rys. 5.2 przedstawiono klasyfikację modelu w zależności od formy opisu zmiennych stanu.



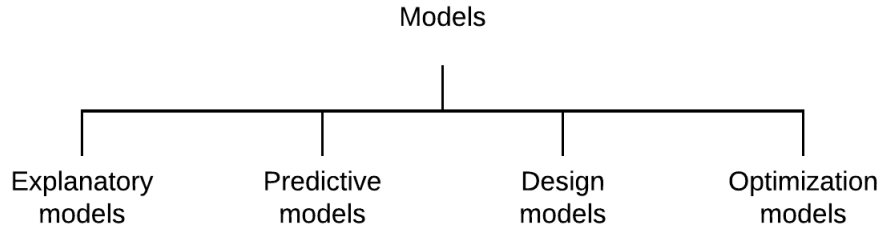
Rys. 5.2. Klasyfikacja modeli w zależności od rodzaju zmiennych stanu [84].

Kolejny zaproponowany podział modeli dotyczy ich przeznaczenia:

- objaśniające (ang. *explantory models*) – pozwalające na zrozumienie zagadnienia
- predykcyjne (ang. *predictive models*) – umożliwiające przewidywanie przyszłych stanów systemu (w różnych konfiguracjach)
- projektowe (ang. *design models*) – służące do analizy różnych możliwości konfiguracyjnych danego środowiska
- optymalizacyjne (ang. *optimization models*) – wykorzystywane do maksymalizacji określonej funkcji celu.

Podział opisanych wyżej modeli przedstawiony został na rys. 5.3.

W przypadku modelowania pojawia się szereg zagadnień związanych z redukcją złożoności w celu osiągnięcia możliwie najprostszej formy pozwalającej na realizację określonych w założeniach celów. Warto zaznaczyć, że stosowanie takich operacji wymaga bardzo dobrej znajomości modelowanego zjawiska, gdyż błędna ocena i nieuwzględnienie określonych elementów modelu może w dalszej perspektywie negatywnie wpływać na wiarygodność. Określenie procesu modelowania przez Shannona (1975), jako bardziej sztukę niż naukę, ma duże uzasadnienie szczególnie w kontekście złożoności i pewnej nieprzewidywalności tego procesu.



Rys. 5.3. Klasyfikacja modeli w zależności od ich przeznaczenia [84].

Symulacja jest procesem pozwalającym na ożywienie modelu w określonych warunkach systemowych, np. w celu oceny realizacji (implementacji) procesów dynamicznych czasu rzeczywistego, z wykorzystaniem danych pochodzących ze świata rzeczywistego. Może też być formą przetwarzania pozwalającą na przewidywanie zachowania systemów (w oparciu o zmiany wektora stanu w dziedzinie czasu). Proces symulacji układów dynamicznych rozpoczyna się od inicjalizacji stanu początkowego, inaczej mówiąc od nadania określonych wartości początkowych zmiennym stanu. W przypadku symulacji komputerowej modele reprezentowane i realizowane są przez programy, natomiast eksperymenty symulacyjne dotyczą wykonywania tych programów dla określonych zbiorów danych.

Modele analityczne w odróżnieniu od procesu symulacyjnego reprezentowane są przez zbiory równań dla których można uzyskać rozwiązanie pozwalające na ilościową lub parametryczną analizę danego procesu. Jednym z przykładów jest optymalizacja MAX, MIN modelu z narzuconymi (m) ograniczeniami [17]:

$$g_i(x) = \begin{cases} \geq 0 \\ 0 \\ \leq 0 \end{cases} \quad \text{dla } i = 1 \ldots \mathrm{m}, \tag{5.1}$$

gdzie x jest wektorem o n współrzędnych (zmiennych $x_j, j = 1 \ldots n$),

$x \in \mathbb{R}_n^+$ (dla wartości rzeczywistych nieujemnych)
$x \in \mathbb{Z}_n^+$ (dla wartości całkowitych), lub
$x \in \mathbb{B}_n^+$ (dla wartości binarnych).

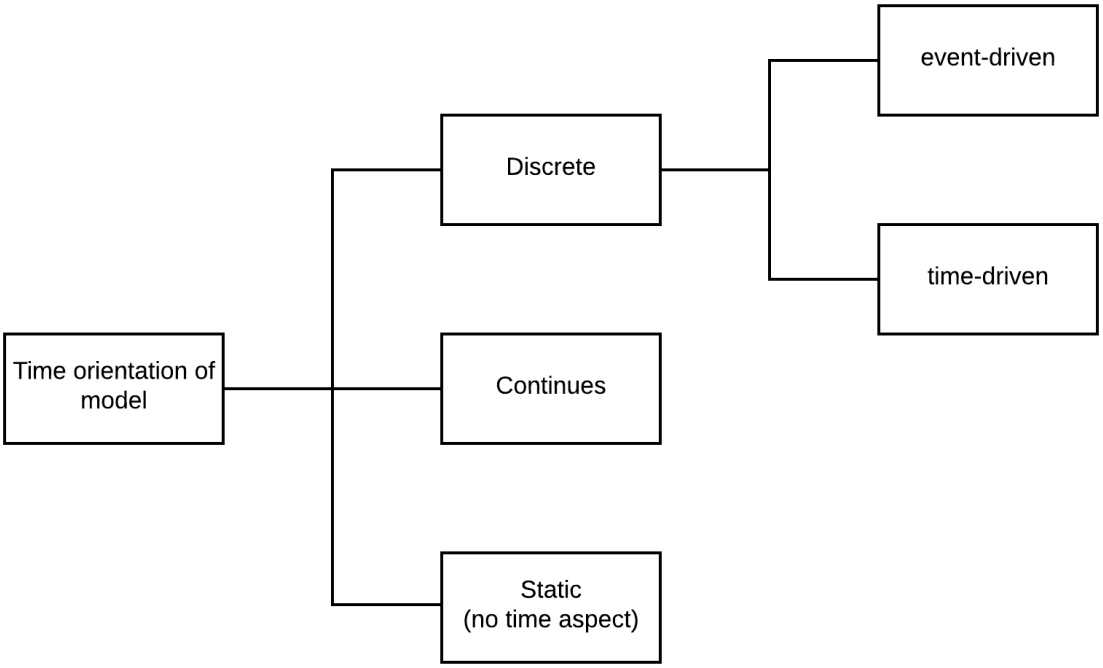
Modele symulacyjne w odróżnieniu od modeli analitycznych, związane są z przeprowadzaniem obliczeń krok po kroku na wartościach zmiennych stanu. Istotne jest określenie poziomu dokładności zmiennych w kontekście prowadzonych kalkulacji. Wymaga to jednak głębokiej znajomości zależności związanych z modelowanym procesem. Nawet czasem twierdzi się [84], że taka forma technik symulacyjnych powinna być rozważana głównie w przypadku, kiedy modele analityczne zawodzą (np. w skutek błędnych założeń dotyczących liniowości modelowanego procesu, zastosowanie zbyt dużych uproszczeń itd.).

Podstawową zaletą modeli symulacyjnych jest łatwość modyfikacji z możliwością wykorzystania w analizie modeli o różnej dokładności oraz alternatywnych konfiguracjach. Idea tworzenia takich programów symulacyjnych jest również bardziej użyteczna dla ekspertów dysponujących nieco mniej zaawansowanym aparatem matematycznym. Złożoność obliczeniowa w przypadku modeli symulacyjnych jest zwykle duża i wymaga odpowiedniego zaplecza sprzętowego. W funkcjonujących systemach symulacyjnych możemy wyróżnić kilka form reprezentacji czasu w wykorzystywanym modelu. Na rys. 5.4 przedstawiony został ogólny podział klasyfikacji modeli według wykorzystywanych form reprezentacji czasu.

Dyskretna forma symulacji zakłada zmiany stanu systemu w ściśle określonych, dyskretnych momentach czasu (oznacza to, że stan systemu nie może się zmieniać pomiędzy poszczególnymi zdarzeniami). Wykorzystywana w takim modelu sekwencja czasu jest odwzorowywana w (skończoną) trajektorię stanów. W dalszej części rozdziału rozważania skupią się wokół dyskretnych form symulacji, które wykorzystane zostały przy budowie środowiska symulacyjnego bazującego na języku SMOL.

5.3. Symulacja dyskretna

Forma zdarzeniowej symulacji dyskretnej została wybrana jako najbardziej właściwa, pozwalająca na przeprowadzenie badań nad sieciami połączonych ze sobą urządzeń pomiarowo-diagnostyczno-sterujących. Tego rodzaju sieciowe modele symulacyjne nadają się bowiem do opisu i implementacji struktur generujących niedeterministyczne rezultaty. Niniejszy podrozdział zawiera studium dotyczące wykorzystania systemów symulacji dyskretnej.



Rys. 5.4. Klasyfikacji modeli według wykorzystywanych form reprezentacji czasu [84].

5.3.1. Dyskretna symulacja zdarzeniowa

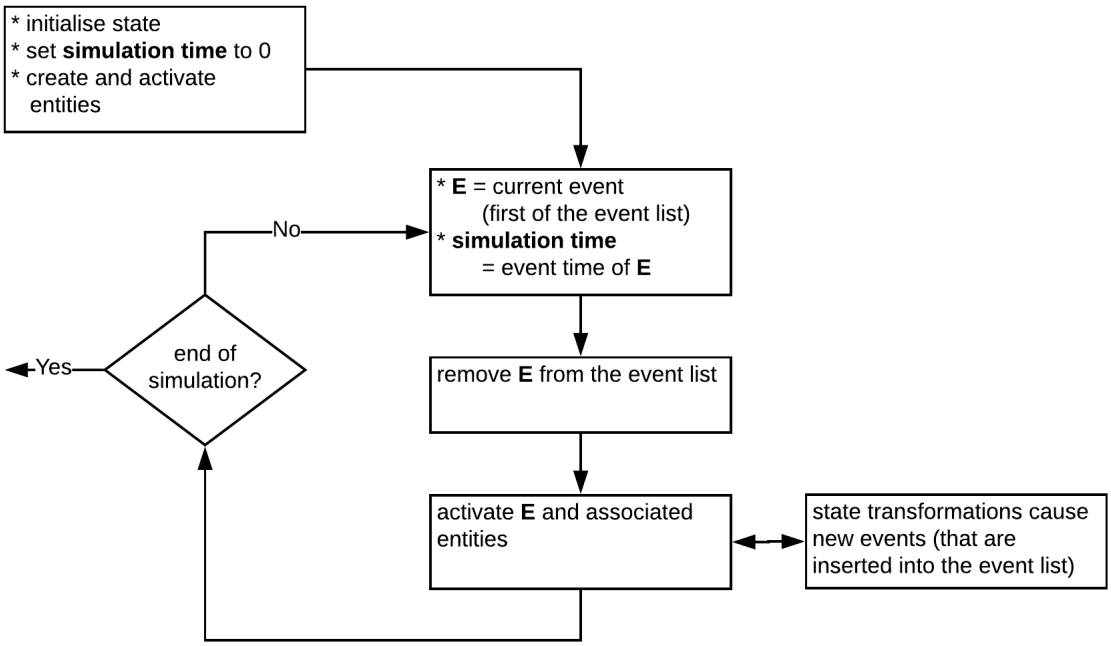
Można uznać, że w modelowaniu procesów opis modelu stanowi statyczną formę reprezentacji struktury systemu rzeczywistego, natomiast jego zachowanie reprezentuje dynamiczną część procesu symulacji. Symulowaną (wewnętrzną) dynamikę procesu reprezentują na zewnątrz przebiegi zmiennych procesu w czasie.

W procesie modelowania dyskretnego mamy do czynienia z dwoma interpretacjami zmiennej niezależnej: czasem rzeczywistym (procesu) oraz czasem modelowanym (może wystąpić tu zjawisko skalowania czasu). Natomiast w symulacji Page'a i Kreutzera [84] czas modelowany stanowi jedynie parametr wewnętrzny modelu, natomiast czas egzekucji programu symulacyjnego odnieść można do czasu rzeczywistego. Oczywiście w tym sensie czas modelu jest fikcyjny i niezależny od rzeczywistego czasu wykonywania symulacji.

Model dyskretnej symulacji jest właściwy w przypadku analizy dyskretnych zdarzeń związanych z np. dystrybucją i dostawą przesyłanych przez sieć pakietów danych. Przykładem zastosowania takich rozwiązań jest symulacja komputerowych systemów sieciowych, jak również modeli sieci telekomunikacyjnych. Na takie wyzwania odpowiada środowisko SMOL, pozwalające na opis struktury (modelowanie) i symulację działania

sieci połączonych urządzeń kontrolno-pomiarowych.

W opisywanej formie symulacji modelowany czas rzeczywisty procesu wyrażany jest interwałami pomiędzy występującymi zdarzeniami. Przejście od jednego zdarzenia do drugiego inicjowane jest poprzez wewnętrzną zmianę stanu symulowanego obiektu. Jak już wspomniano, pomiędzy zdarzeniami nie są generowane żadne zmiany dotyczące funkcjonowania modelu. Symulacja zdarzeniowa zakłada, że każde zdarzenie przechowuje informację na temat typu zdarzenia, zdarzeń powiązanych oraz czasu, w którym zostało zainicjowane. W przypadku wystąpienia zdarzenia, czas modelowany jest automatycznie (przyrostowo) aktualizowany czasem zainicjalizowanego zdarzenia, a w konsekwencji wykonywane są transformacje na zmiennych stanu. Wewnętrzna zmiana stanu modelu, wynikająca z przeprowadzonych transformacji, prowadzi do zainicjalizowania kolejnych zdarzeń (reakcja łańcuchowa). Taki sposób postępowania przedstawiono na rys. 5.5.



Rys. 5.5. Schemat dyskretnej symulacji zdarzeniowej [84].

Proces zdarzeniowej symulacji dyskretnej polega na iteracji wzdłuż listy zdarzeń (ang. *event list*), czyli wykonywania poszczególnych zdarzeń w określonej kolejności do momentu opróżnienia listy lub zakończenia procesu symulacyjnego wcześniej (ze względu na spełnienie określonego kryterium). W przypadku, gdy kilka zdarzeń inicjalizowanych jest w tym samym momencie czasu (symulacja zdarzeń równoległych) w systemach symulacyjnych wykonywane są one nadal w oparciu o listę zdarzeń. Jednym z popularnych

form obsługi (listy) zdarzeń jest zastosowanie modelu FCFS[1] (FIFO) lub wykorzystanie priorytetów przypisanych do poszczególnych zdarzeń.

Według autorów [84] dyskretne systemy zdarzeniowe oferują trzy różne formy synchronizacji zmiany stanu modelu i (upływającego) czasu.

Pierwszą z nich jest wykorzystanie zdarzeń, które charakteryzują zmiany stanu w określonym momencie czasu modelowanego. Zdarzenia są wówczas generowane poprzez transformację stanu modelu (czas zdarzeń może być deterministyczny lub stochastyczny).

Drugą formą jest wykorzystanie aktywności, opisywanych jako grupa operacji wykonywanych w określonych przedziałach czasowych. Zmiany stanu modelu realizowane są wówczas na początku i końcu aktywności. Przykładem może być symulacja serwerowego systemu kolejkowego, gdzie mamy do czynienia z gromadzeniem danych (początek aktywności) oraz ich publikowaniem (koniec aktywności).

Trzecią formą synchronizacji jest wykorzystanie metodologii procesowej, która związana jest wykonywaniem pewnych aktywności wpisanych w cykl życia modelu. Jednym z przykładów takiej formy synchronizacji jest wykonywania operacji maszynowych w symulacji procesu produkcji (np. sekwencja zdarzeń o różnej długości czasu przetwarzania).

Relacje pomiędzy poszczególnymi podejściami przedstawione zostały na rys. 5.6 a dla zobrazowania różnic przedstawiono model systemu realizacji zamówień (ang. *orders*).

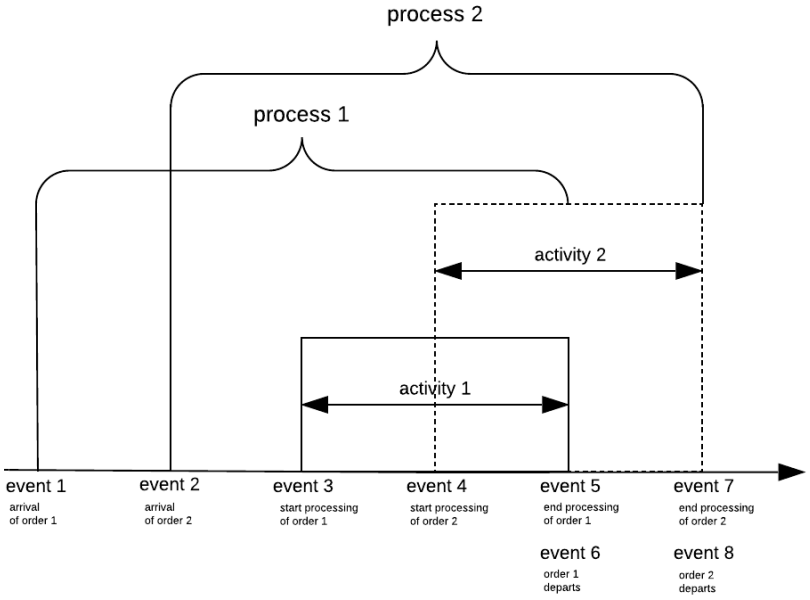
Forma symulacji dyskretnej jest zaawansowanym narzędziem komputerowym pozwalającym na prowadzenie analizy związanej z działaniem i optymalizacją złożonych systemów. Na koniec warto podkreślić, że jakość symulacji uzależniona jest od danych pochodzących z analizowanych systemów rzeczywistych. Im jakość pozyskanych danych jest wyższa, tym zachowanie modelowanego systemu bardziej odpowiada jego rzeczywistemu odpowiednikowi.

5.4. System symulacji dyskretnej oparty na języku SMOL

Przedstawiony ogólnie opis w rozdziale 4 języka SMOL zawiera jedynie elementy pozwalające na charakterystykę struktury systemów z rodziny PDS. Wynikiem działania parsera jest reprezentacja sieci podana w formie grafu, zapisanego do pliku wyjściowego XML. Dodatkowym wynikiem jest również diagram w postaci graficznej, który wizualizuje strukturę połączonych elementów wchodzących w skład analizowanej sieci.

W ramach prac rozwojowych zdecydowano się na rozwinięcie tego narzędzia i wyposażenie go w mechanizmy pozwalające przeprowadzać symulację działania sieci PDS (poza usystematyzowanym opisem jej struktury). Jak już wspomniano, wybór formy dyskretnej symulacji podyktowany jest rodzajem uwzględnianej dziedziny systemów, a w szczególności sieci pomiarowych, komputerowych i telekomunikacyjnych.

[1] ang. *First come, first served.*



Rys. 5.6. Relacje pomiędzy zdarzeniami, aktywnościami i procesami w przypadku systemu ze zdefiniowaną kolejnością przetwarzania [84].

W implementacji systemu symulacji dyskretnej zastosowano stworzoną specjalnie do tego celu ramę programową (ang. *freamwork*) DESMO-J[2], która powstała na wydziale informatyki uniwersytetu w Hamburgu, pozwalającą na wykonywanie symulacji w oparciu o język Java[3].

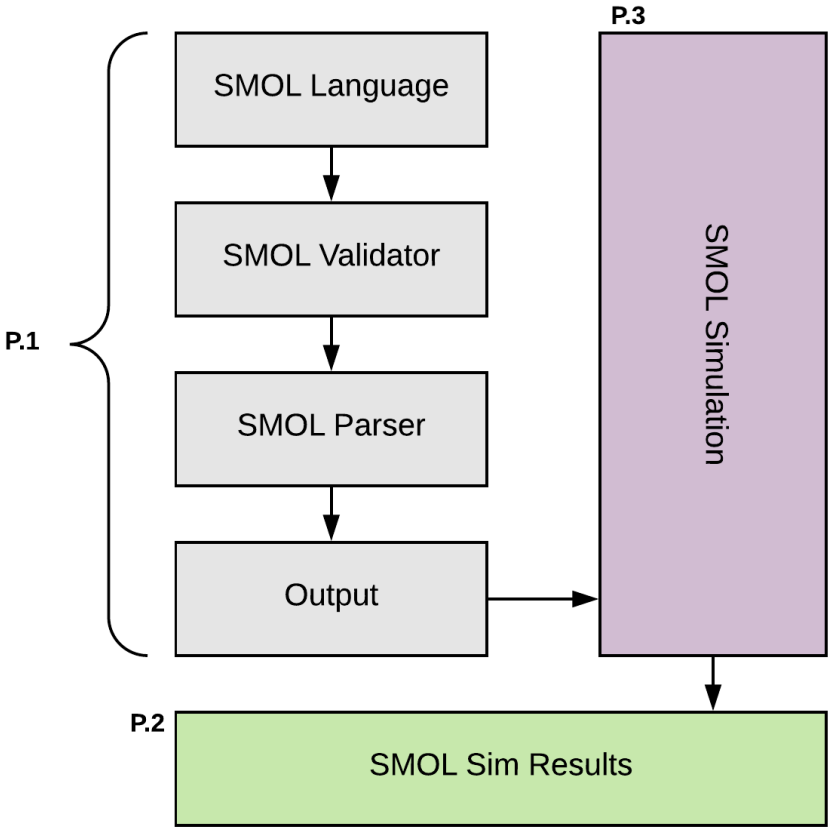
5.4.1. Wykorzystanie języka SMOL jako narzędzia domenowego

SMOLSim jest symulacyjną platformą składającą się z kilku połączonych ze sobą modułów programistycznych. Każdy z nich odpowiada za realizację określonej funkcji. Celem prac implementacyjnych było opracowanie możliwie uniwersalnej platformy, która pozwala też na łatwą modyfikację i rozbudowę realizowanego projektu.

Struktura elementów programowych wchodzących w skład utworzonej aplikacji przedstawiona została na rys. 5.7.

Jak już wspomniano, w początkowej fazie język SMOL wykorzystywany był jedynie do opisu struktury sieci oraz wygenerowania obiektu XML zawierającego zdefiniowany mo-

[2]http://desmoj.sourceforge.net
[3]https://www.oracle.com/java/index.html



Rys. 5.7. Schemat architektury rozwiązania SMOL.

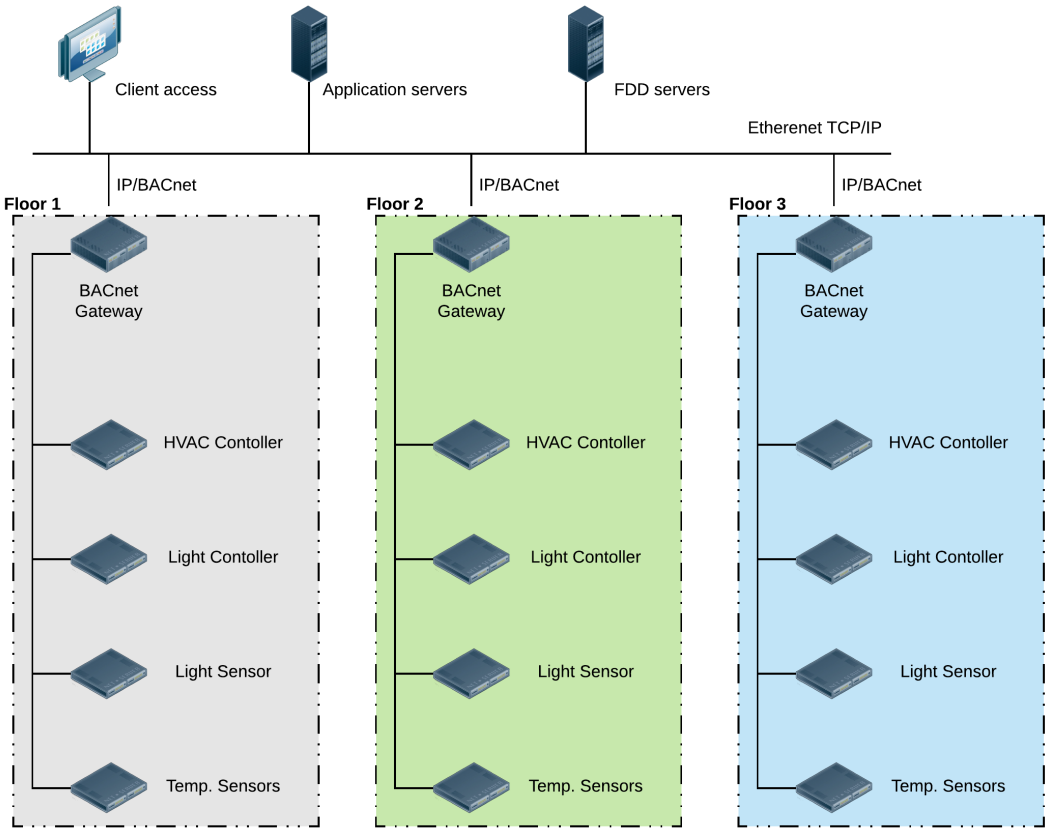
del połączeń. Na rys. 5.7 fragment ten oznaczony został na rysunku symbolem **P.1**. Proces przetwarzania opisu struktury sieci rozpoczyna się od analizy kodu SMOL (walidacja) w celu określenia poprawności składni, po czym następuje parsowanie polegające na wyszukiwaniu funkcji oraz ich argumentów odpowiedzialnych za wywoływanie konkretnych akcji (np. rozpoczęcie procesu symulacji lub wygenerowanie graficznej reprezentacji struktury sieci).

Wygenerowana w ten sposób struktura trafia do modułu symulacyjnego (SMOL Simulation) i tam poddawana jest dalszemu przetwarzaniu polegającemu na zapewnieniu kompatybilności pomiędzy kodem programu napisanym w SMOL a narzędziem symulacji dyskretnej DESMO-J.

5.5. Generowanie programu symulacyjnego w języku SMOL

Przykład pokazujący użyteczność opracowanego środowiska SMOLSim polega na przeprowadzeniu eksperymentów symulacyjnych prezentujących działanie sieci PDS służącej do wymiany informacji pomiędzy urządzeniami pomiarowo-wykonawczymi za pomocą standardu TCP/IP oraz BACnet (rozdział 2.2.9).

Stworzona w oparciu o protokoły komunikacyjne (TCP/IP oraz BACnet) sieć PDS może mieć dowolnie złożoną strukturę. Rodzaje wykorzystywanych mediów komunikacyjnych przedstawione i scharakteryzowane zostały w rozdziale **??**. W omawianym eksperymencie wykorzystano standard Ethernet jako warstwę komunikacyjną, która zapewnia możliwość szybkiej integracji z innymi urządzeniami lub aplikacjami komputerowymi (np. program komputerowy, działający w sieciach LAN, wspomagający proces podejmowania decyzji na podstawie danych pomiarowych lub zwykłe urządzenia pomiarowe, itd.).



Rys. 5.8. Schemat działania sieci PDS.

Tablica 5.1: Opis oznaczeń zastosowanych na rys. 6.2

Oznaczenie	*Opis*
bacnet-floor(n)-gw	Konwerter pomiędzy urządzeniami BACnet a siecią TCP/IP
hvac-bacnet-floor(n)	Sterownik systemu HVAC (BACnet) umieszczony na piętrze n
light-bacnet-floor(n)	Czujnik oświetlenia (BACnet) umieszczony na piętrze n
temp-sensor-bacnet-floor(n)	Czujnik temperatury (BACnet) umieszczony na piętrze n

Platforma SMOLSim została zaprojektowana z myślą budowania systemów symulacji pozwalających na odwzorowanie mechanizmów wykorzystywanych do komunikacji pomiędzy urządzeniami automatyki budynkowej, zaś standard BACnet jest jednym z najczęściej stosowanych rozwiązań przy projektowaniu i wdrażaniu systemów automatyki budynkowej w Stanach Zjednoczonych.

Przykład zastosowania platformy SMOL do modelowania i symulacji został zaprezentowany na rys. 6.1 i 6.2. Dotyczy on symulacji działania sieci PDS wykorzystywanej w trzypiętrowym budynku. Każde z pięter zostało wyposażone w urządzenia pomiarowe (czujniki temperatury i wilgotności) oraz wykonawczo-pomiarowe (system HVAC). Na pierwszym rysunku (6.1) widzimy koncepcję działania sieci pomiarowej wykorzystującej BACnet oraz TCP/IP do komunikacji pomiędzy poszczególnymi urządzeniami, drugi (6.2) zaś prezentuje opisaną sieć w postaci grafu, wygenerowanego w środowisku SMOL. Graf jest formą reprezentacji sieci umożliwiającą optymalizację, modelowanie i symulację.

Kod programu w języku SMOL opisujący strukturę zaprezentowanej sieci PDS umieszczono w dodatku A. Natomiast opis poszczególnych skrótów wykorzystanych na rys. 6.2 umieszczony został w tabeli 5.1.

Dla graficznej reprezentacji sieci PDS z rys. 6.2 przeprowadzono symulację ruchu pakietów z uwzględnieniem struktur sieci: LAN stanowiącej medium do przesyłania pakietów TCP oraz BACnet związanej z automatyką budynkową (urządzenia pomiarowe, wykonawcze i sterujące).

W symulacji przyjęto założenie, że wielkość ramki TCP przesyłanej przez sieć wynosi 1518 bajtów, zaś czas transmisji danych zawartych w ramce przesyłanej z szybkością 1 [Mb/s], wynosi:

$$T_d = 0,012144[s] = 12\ 144\ [\mu s] \tag{5.2}$$

Liczba próbek **L** przetwarzanych przez system pomiarowy w jednostce czasu, jest nie większa niż odwrotność czasu Td. Największa liczba próbek przetwarzanych przez

system w czasie 1[s] wynosi:

$$L = 1/T_d = 82,3\ [S/s] \tag{5.3}$$

Prędkość rozchodzenia się fali elektromagnetycznej w próżni, w tym prędkość światła, wynosi $3 * 10^8$ [m/s] i jest to graniczna prędkość rozchodzenia się sygnału w naturze. Dla linii pomiarowej wykonanej z kabla elektrycznego prędkość rozchodzenia się fali elektromagnetycznej jest mniejsza niż $3 * 10^8$ [m/s]. Szybkość rozchodzenia się fali elektromagnetycznej w kablu jest zależna od jego jakości i wynosi około $2 * 10^8$ [m/s]. Czas Tp propagacji sygnału na odcinku 1000 [m] wynosi zatem:

$$T_p = l/V = 1000/(2 * 10^8) = 5\ [\mu s] \tag{5.4}$$

W symulacji złożono, że czas transmisji ramki danych o objętości 1518 bajtów wynosi:

$$T_s = T_d + T_p = 12\ 149\ [\mu s] \tag{5.5}$$

Przy powyższych założeniach prowadzono eksperyment symulacyjny z czasem symulacji 30 min. Wyniki prowadzonej symulacji zilustrowano na rys. 6.3.

Wykresy te pokazują jak w czasie kształtuje się ruch pakietów z danych przenoszących informację pomiarową do systemu wnioskowania i analizy (oznaczonego na rys. 6.2 jako FDD-Server). Wykresy po lewej pokazują czas przesyłania informacji pomiarowej pochodzącej z czujników i aktuatorów, poprzez konwerter interfejsów (BACnet - TCP) do systemu komputerowego (FDD-Server). Widać, że czasy przesyłu danych mieszczą się w podobnym przedziale.

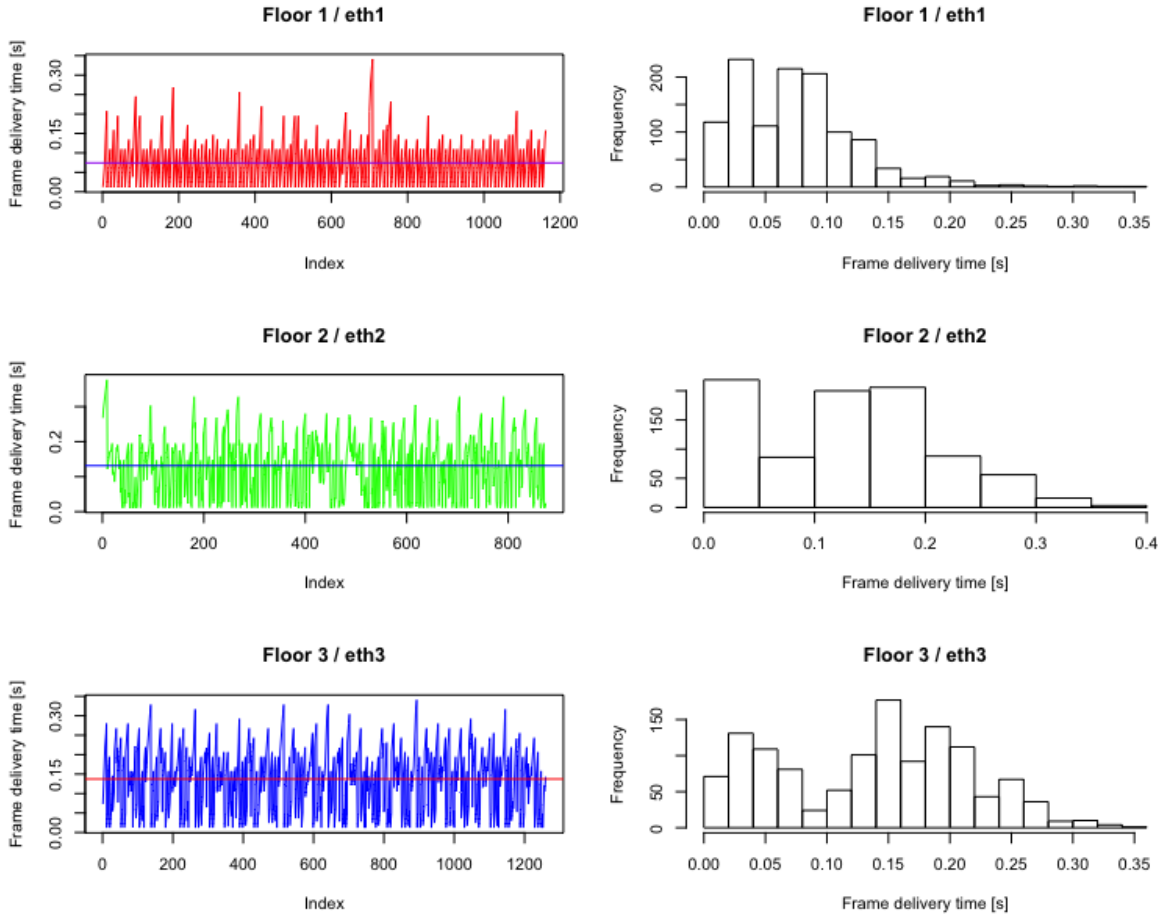
Średnia liczba pakietów z danymi przekazywana przez sieć w symulowanej chwili czasu kształtuje się na poziomie około 44 pakietów (przy możliwej maksymalnej liczbie 83). Sieć w takiej konfiguracji obciążona jest zatem jedynie w około 50%.

Dla obciążeń od 50% do 80% średni czas odpowiedzi w sieciach LAN wzrasta do 0.1 [s] (aplikacje transmitujące dane z wykorzystaniem WWW i FTP działają nadal poprawnie). Dla stałych obciążeń powyżej 80% sieć **nie zapewnia** poprawnego działania (bez zauważalnych opóźnień). Taka sieć kwalifikuje się do rekonfiguracji [70].

W przedstawionym eksperymencie zilustrowano podstawowe możliwości środowiska symulacyjnego SMOLSim. Bardziej złożone eksperymenty zaprezentowane zostaną w rozdziale 6.

5.5.1. Translacja kodu na język środowiska symulacyjnego

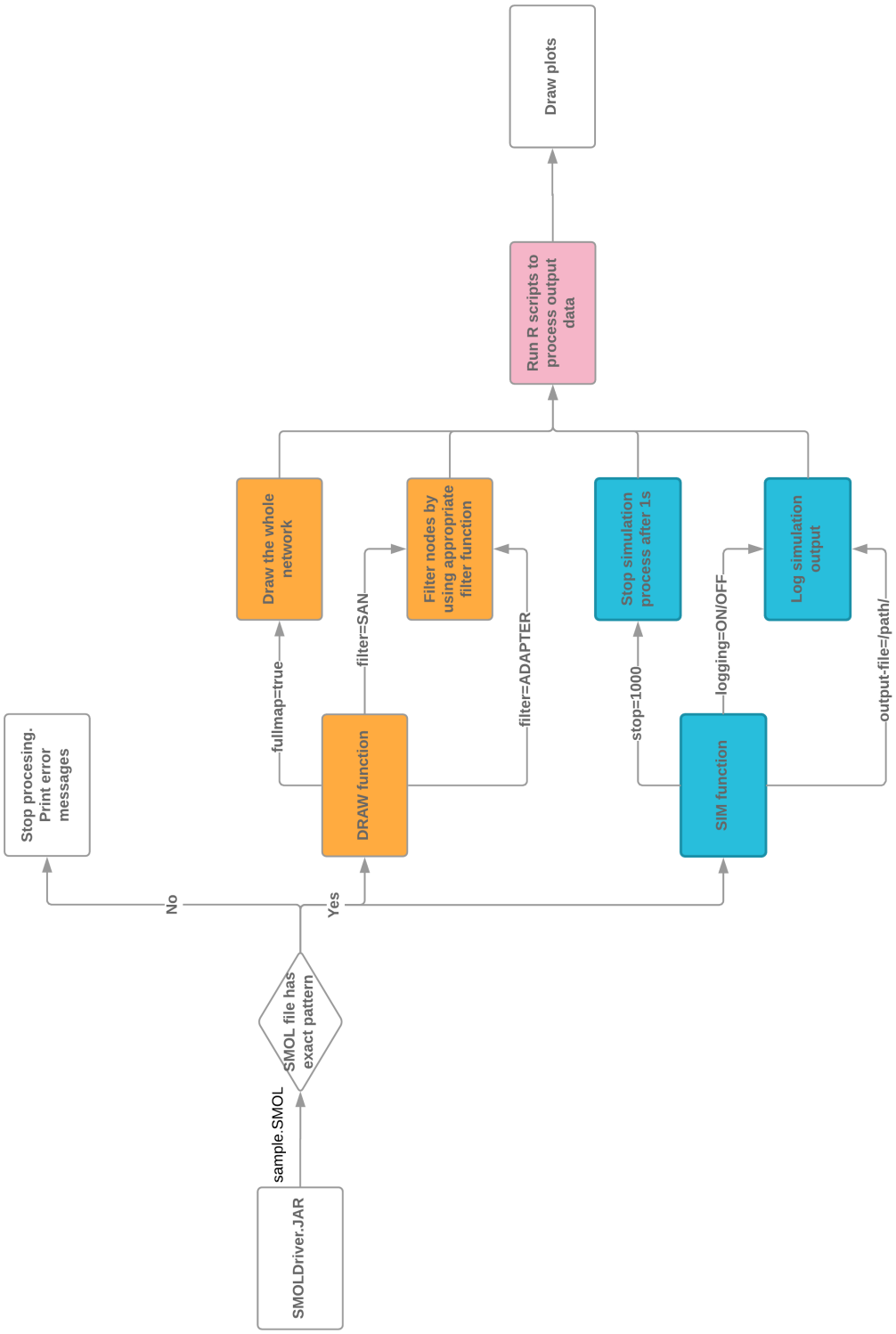
Sposób translacji kodu w języku SMOL (opisującego strukturę sieci PDS) na postać zrozumiałą dla środowiska symulacyjnego przedstawiony został na rys. 5.11.



Rys. 5.10. Ruch pakietów przenoszących informację pomiarową do systemu wnioskowania i analizy.

Pierwszy etap polega na analizie syntaktycznej kodu, gdzie weryfikowana jest poprawność zapisu poprzez sprawdzenie spełnienia kryteriów niezbędnych do dalszego przetwarzania. Jeśli proces analizy syntaktycznej zakończy się niepowodzeniem dalsze działania zostaną zatrzymane.

Programista języka SMOL po dokonaniu opisu sieci PDS może wybrać dwa rodzaje akcji. Pierwsza z nich dotyczy wywołania funkcji *draw()*, która odpowiada za analizę opisanej struktury oraz wygenerowanie graficznej reprezentacji w postaci grafu. Przykład wygenerowanego grafu, opisującego sieć połączeń, umieszczono na rys. 6.2.



Rys. 5.11. Przetwarzanie kodu w środowisku SMOL.

5.6. Narzędzia wspomagające budowę systemów symulacji

Powstało wiele narzędzi wspomagających budowę systemów symulacyjnych. W ramach prowadzonych badań, wybrane zostały dwa główne środowiska. Pierwsze (uruchamiane na platformie JVM) stworzono z wykorzystaniem języka Java (DESMO-J) oraz drugie opracowano w środowisku języka skryptowego Python (SimPy). W dalszej części pracy przedstawione i scharakteryzowane zostaną oba narzędzia.

5.6.1. DESMO-J – charakterystyka i opis biblioteki

DESMO-J (ang. *Discrete-Event Simulation and MOdelling in Java*) jest zaawansowanym środowiskiem symulacyjnym opracowanym z wykorzystaniem obiektowego języka programowania jakim jest Java. Głównym celem towarzyszącym powstaniu tego środowiska[4] jest wspieranie dyskretnej symulacji procesowo-zdarzeniowej.

DESMO-J stanowi zatem rozszerzenie środowiska Java o funkcje ułatwiające budowanie dyskretnych środowisk symulacyjnych. Programista takich systemów otrzymuje następujący zbiór funkcji:

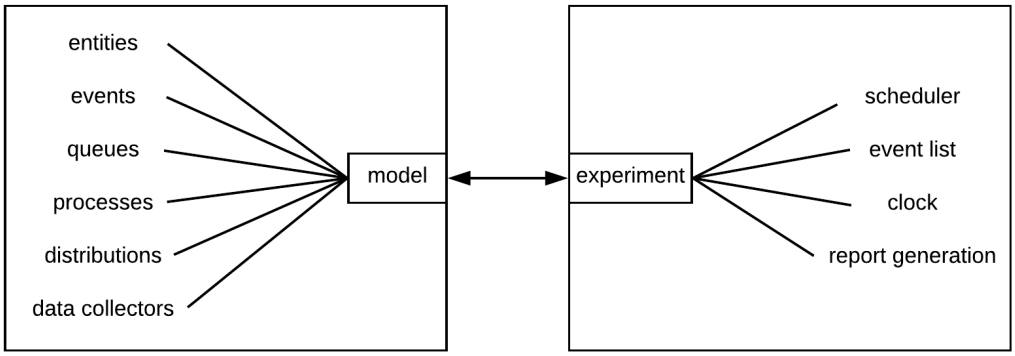
- gotowe do użycia typowe elementy wykorzystywane w modelowaniu (kolejki, generatory liczb pseudolosowych, kolektory danych)
- zbiór klas abstrakcyjnych umożliwiający nadanie określonej roli implementowanym obiektom (np. model, zdarzenie, process)
- przygotowane środowisko eksperymentu pozwalające na harmonogramowanie zadań, zarządzanie listą zdarzeń i czasem symulacji oraz śledzeniem przebiegu symulacji i późniejsze generowanie raportów.

Rama programowa (ang. *framework*) DESMO-J udostępnia mechanizm separacji modelu od eksperymentu, który pozwala na wykorzystanie tego samego modelu w różnych eksperymentach. Można w takich warunkach symulacyjnych zweryfikować, która wersja modelu lub która grupa modeli daje najlepsze wyniki. Na rys. 5.12 przedstawiono podział ze względu na funkcje sprawowane przez model oraz eksperyment.

5.6.2. SimPy – charakterystyka i opis biblioteki

SimPy jest darmowym środowiskiem programistycznym stworzonym do dyskretnej symulacji procesowo-zdarzeniowej. Narzędzie zostało zaimplementowane w języku Python,

[4]Narzędzie to powstało i jest nadal rozwijane na katedrze systemów komputerowych Uniwersytetu w Hamburgu.



Rys. 5.12. Reprezentacja modelu i eksperymentu w środowisku DESMO-J.

co jest jego niewątpliwą zaletą. Technologiczny próg dla wejścia w to środowisko nie jest wysoki i pozwala nawet początkującemu programiście stosunkowo szybko budować własne implementacje [45].

Zachowanie poszczególnych komponentów (np. urządzeń sieciowych, serwerów, sensorów) jest modelowane z wykorzystaniem procesów, które rejestrowane są za pomocą zmiennej środowiskowej (ang. *environemnt*). Procesy te mogą wchodzić między sobą w interakcje poprzez wykorzystanie mechanizm zdarzeń (ang. events).

W środowiskach symulacyjnych powszechne stosuje się mechanizm wątków (ang. threads) jako sposoby metoda procesów, natomiast w SimPy wykorzystuje się koncepcję opierającą się na generatorach. Są to wewnętrzne funkcje środowiska Python pozwalające programiście na określenie, że dana funkcja może być przerwana (zakończona, a następnie wznowiona, np. z wykorzystaniem mechanizmu zdarzeń). Określenie wyjścia oraz miejsca powrotu realizowane jest w środowisku Python za pomocą funkcji *yield*. Każde następne odwołanie się do takiej funkcji powoduje jej natychmiastowe i automatyczne wznowienie w miejscu przerwania.

Wynik działania procedury 1 przedstawiono za pomocą procedury 2.

W procedurze 1 zademonstrowano sposób wykorzystania opisanego wcześniej mechanizmu funkcji *yield*. Jak widać na załączonym przykładzie wywołanie powoduje zatrzymanie działania programu na 5 [s] a następnie wznowienie jazdy na 2 [s]. (w miejscu przerwania).

Tak zdefiniowania powyżej postać opisu procesów pozwala stosunkowo łatwo budować nowe środowisko symulacyjne. Stosunkowo prosta syntaktyka oraz zrozumiała semantyka tego języka sprawiają, że jest on często wybierany do implementacji projektów naukowych, inżynieryjnych i studenckich.

Procedura 1 Przykładowy program w języku Python, środowisko SimPy [45].

1: **function** CAR(environment)

```
...     while True:
...         # The process requires a reference to an environment
...         # in order to create new events.
...         # The current simulation time is returned by
...         # the environment.now function.
...         print('Start parking at %d' % environment.now)
...         parking_duration = 5
...
...         # Environment.timeout() function creates a Timeout event.
...         # The event describes the point in time the car is done
...         # parking.
...         # By yielding the event, it signals the simulation that
...         # it wants to wait for the event to occur.
...         yield environment.timeout(parking_duration)
...
...         print('Start driving at %d' % environment.now)
...         trip_duration = 2
...         yield environment.timeout(trip_duration)
```

2: **end function**

5.7. Planowy rozwój środowiska SMOLSim

Platforma symulacyjna jak również sam język SMOL do opisu struktury modelowanych sieci PDS daje duże możliwości w kontekście symulacji i optymalizacji syntezowanej struktury. Język SMOL wzbogacany jest o nowe funkcje rozszerzające jego możliwości. Następne prace dotyczą budowy modułu symulatora pozwalającego w uniwersalny sposób wizualizować wyniki działania. W planach jest również implementacja wykorzystywanych w automatyce obiektowej protokołów komunikacyjnych. Opracowywana jest również metoda komunikacji z symulatorem poprzez aplikacje sieciowe (www).

5.8. Podsumowanie

Zaletą rozwijanego środowiska badawczego opartego na języku SMOL oraz stowarzyszonym środowisku symulacyjnym jest możliwość dogodnego opisu, wizualizacji oraz prowadzenia symulacji analizowanej sieci pomiarowo-diagnostyczno-sterującej (PDS). Projektant takiej sieci otrzymuje funkcjonalne narzędzie pozwalające sprawnie opisywać strukturę systemu lub monitorowanego obiektu oraz ustawiać związane z nim parametry, ogra-

Procedura 2 Wynik działania procedury 1 w środowisku Python [45].

```
>>> # import required libs
>>> import simpy

>>> # create an instance of Environment
>>> env = simpy.Environment()

>>> # Environment (env) instance is passed into car process function.
>>> # A process generator needs to be started and added to the
>>> # environment via env.process() function.
>>> env.process(car(env))
<Process(car) object at 0x...>

>>> # start the simulation by calling run() and passing an end
>>> # time (finish program after 15 sek).
>>> env.run(until=15)
Start parking at 0
Start driving at 5
Start parking at 7
Start driving at 12
Start parking at 14
```

niczenia i relacje, jak również sprawnie przeprowadzać symulacje.

Platformę SMOL stworzono w celu rozwiązywania problemów charakterystycznych dla dowolnych rozłożonych systemów sieciowych. W kolejnym rozdziale zaprezentowano przykłady wykorzystania platformy SMOL w celu potwierdzenia jej uniwersalności i użyteczności.

6

PRZYKŁADY WYKORZYSTANIA OPRACOWANYCH ROZWIĄZAŃ

Rozdział opisujący praktyczne wykorzystanie języka SMOL oraz środowiska SMOLSim. W pierwszej części pracy prezentowane są założenia oraz wyniki dotyczące inteligentnego systemu do zarządzania oświetleniem autostrad (badania zlecone na potrzeby zewnętrznej firmy). W pozostałej części rozdziału zaprezentowane zostaną założenia oraz wnioski wynikające ze zrealizowanych oraz obecnie realizowanych prac dyplomowych opartych na zaprojektowanym środowisku symulacyjnym. Przedstawiony zostanie również aspekt dydaktyczny zbudowanego środowiska oraz możliwości wykorzystania go w studenckich projektach badawczych.

6.1. Wprowadzenie

Rozwój inteligentnego budownictwa w ostatnich latach sprawił, że znacznie powiększyły się możliwości wykorzystania platformy SMOL do modelowania i symulacji sieci PDS. Inteligentne budownictwo jest obecnie synonimem produktywnego i ekonomicznie efektywnego środowiska, które umożliwia optymalizację elementów: systemów, struktur, usług, zarządzania, a także wewnętrznych zależności pomiędzy tymi elementami [46]. Projektowane są coraz bardziej złożone systemy optymalnego zarządzania systemami cieplnymi w budynkach, poprzez wykorzystanie ogrzewania oraz chłodzenia pasywnego [62, 63]. Powstają również hybrydowe systemy energetyczne wykorzystujące odnawialne źródła energii. Inteligentne budownictwo to również systemy alarmowe, sieci monitoringowe, systemy przeciwpożarowe oraz systemy kontroli dostępu. Rozwiązania takie wykorzystują

wspólną sieć do komunikacji pomiędzy poszczególnymi modułami. Platforma SMOL zostanie tu wykorzystana jako narzędzie symulacji i optymalizacji działania rozłożonych sieci pomiarowo wykonawczych [61], ze szczególnym uwzględnieniem systemów automatyki obiektowej i budynkowej.

6.2. Środowisko SMOL w projektowaniu inteligentnego oświetlenia

W niniejszym rozdziale przedstawione zostanie praktyczne zastosowanie platformy SMOL pozwalające na opracowanie i weryfikację metody oszczędności oświetlenia na drogach szybkiego ruchu. Opisany zostanie przedmiot analizy, metoda symulacji oraz wyniki badań wraz z oceną przydatności proponowanego rozwiązania.

Na rys. 6.1 przedstawiono przykład omówionej dalej sieci zarządzającej oświetleniem autostrad wygenerowanej za pomocą parsera SMOL. Kod programu w języku SMOL opisujący strukturę zaprezentowanej sieci PDS umieszczony został w dodatku C.

6.3. Przedmiot symulacji

Rozważmy praktyczny przykład rozproszonego systemu PDS realizującego koncepcję inteligentnego systemu oświetlenia drogowego.

W analizowanym przykładzie lampy mogą komunikować się między sobą informując o wykryciu nadjeżdżającego pojazdu. Celem tego badania jest też znalezienie odpowiedzi na pytanie, czy warto wykorzystać - biorąc pod uwagę przewidywane oszczędności - projektowany system PDS do autonomicznego zarządzania oświetleniem drogowym.

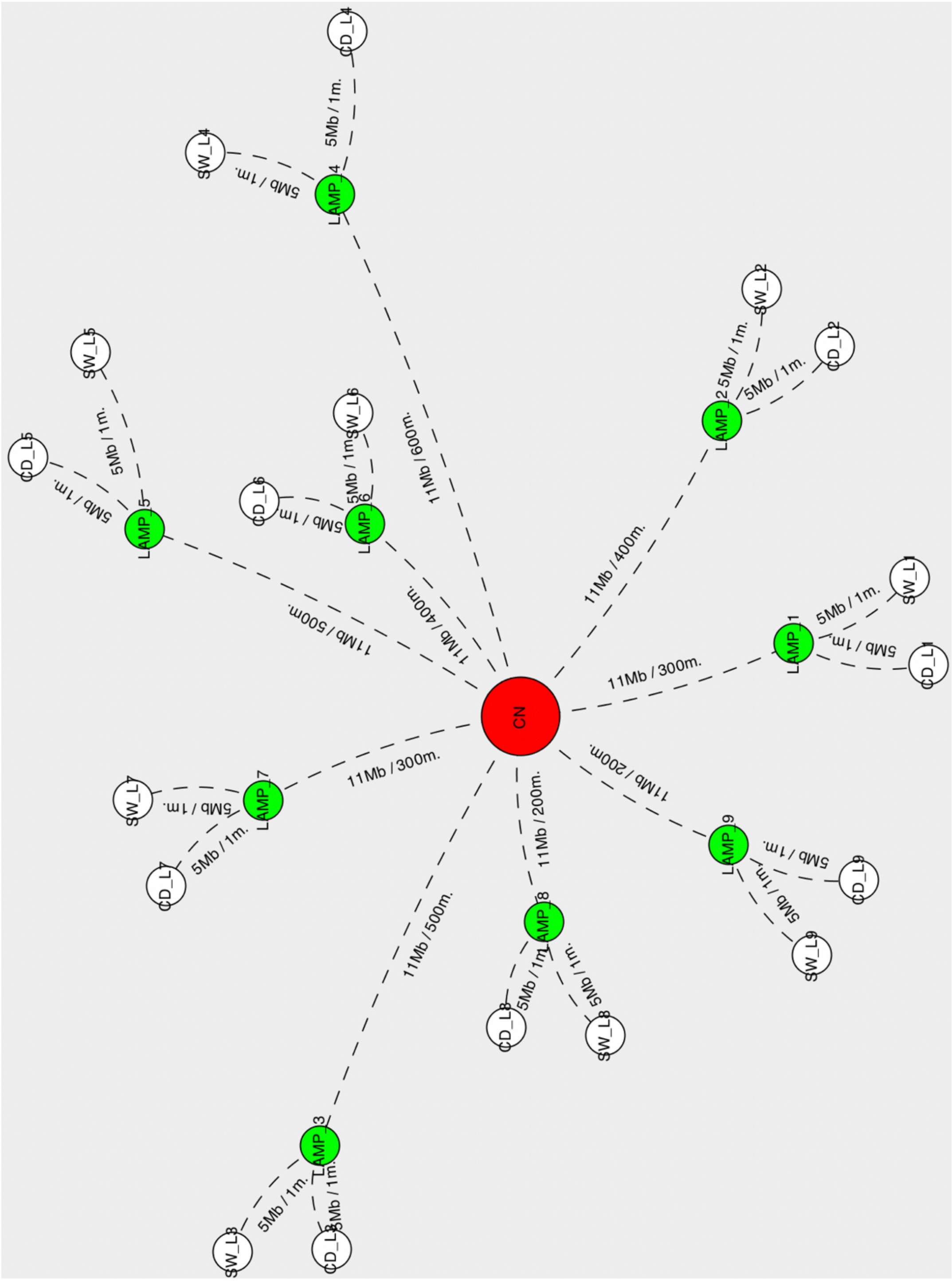
Przedstawione scenariusze testowe mają zatem na celu wykazanie możliwości skutecznej implementacji oświetleniowego systemu PDS oraz weryfikacji (potencjalnej) opłacalności energetycznej proponowanego rozwiązania.

Schemat sytuacyjny będący podstawą scenariuszy oszczędzania energii oświetlenia drogowego rozważany w przeprowadzanych symulacjach przedstawiono na rys. 6.2.

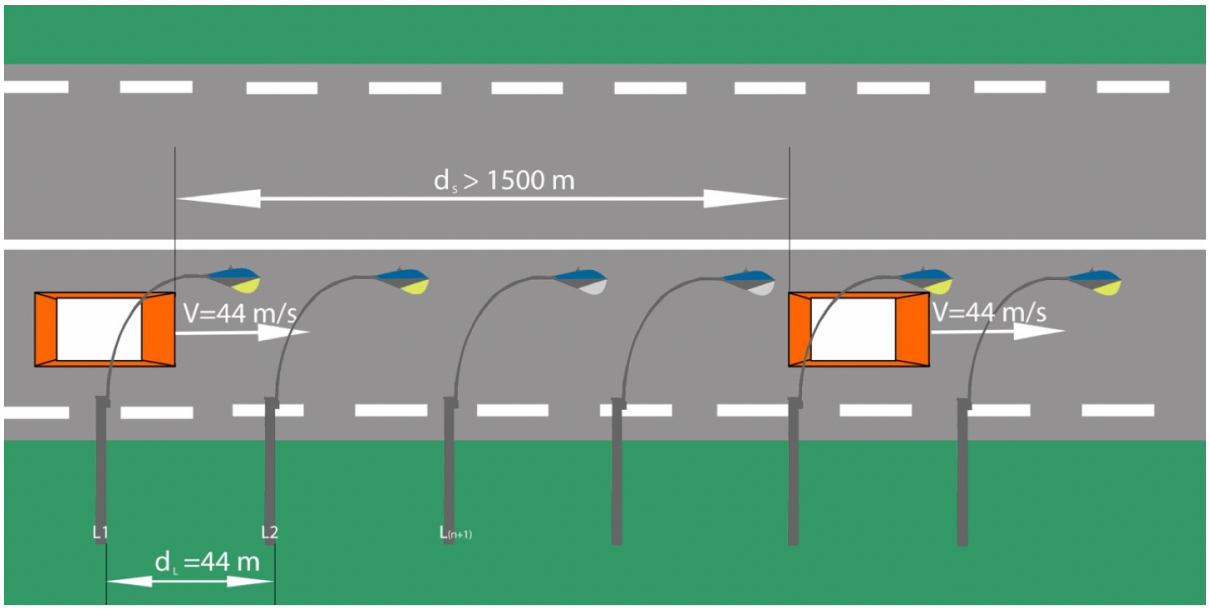
6.4. Scenariusze badawcze

Zaprezentujemy tu dwa scenariusze testowe. Poprzez testowanie projektowanej sieci staramy się wskazać wady i zalety rozpatrywanego rozwiązania.

Przed rozpoczęciem fazy testowania należy przyjąć założenia pozwalające na sprecyzowanie scenariuszy prowadzenia symulacji. Zakładamy mianowicie, że wszystkie samochody poruszają się w jednakowych odstępach ze stałą prędkością 160 [km/h] tj. 44



Rys. 6.1. Graficzna reprezentacja struktury sieci zarządzającej oświetleniem drogowym jako wynik parsowania SMOL.



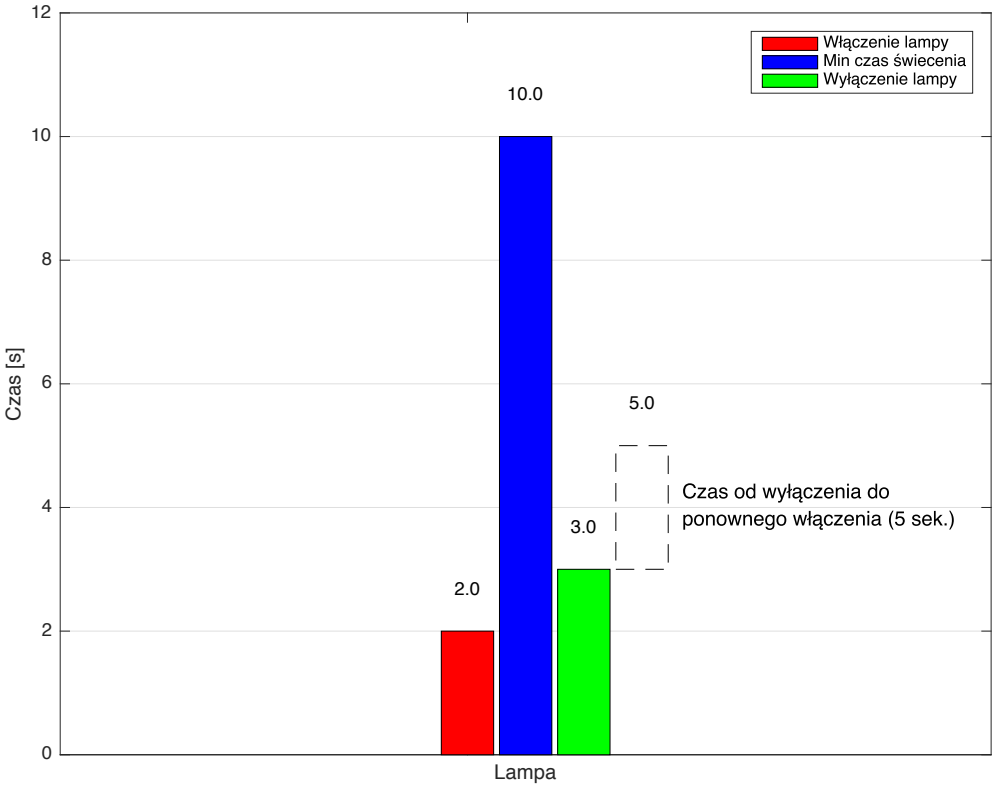
Rys. 6.2. Schemat sytuacyjny analizowanej symulacji drogowej.

[m/s]. Ponadto założono, że odległości pomiędzy lampami wynoszą 44 [m], czas włączenia lampy wynosi 2 [s]. Minimalny czas świecenia lampy definiujemy jako 10 [s] oraz czas od wyłączenia do ponownego włączenia - to 5 [s]. Cykl świecenia lampy zaprezentowany został na rys. 6.3. Określamy również sposób włączania się lamp względem poruszającego się samochodu ustalając, że podczas wykrycia auta włącza się lampa aktualna, której sensor wykrył obecność samochodu (trzecia w przód względem lampy aktualnej w kierunku jazdy), co pokazano na rys. 6.4. Dla zachowania warunków bezpieczeństwa jazdy przy założonej prędkości 160 [km/h] przyjmujemy, że przy wjeździe na oświetloną drogę zapalają się jednocześnie (i inicjalnie) cztery lampy[1].

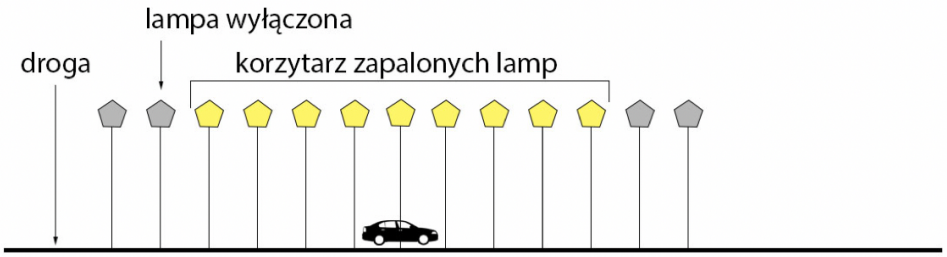
Schemat reprezentujący konfigurację sprzętową poddawanego analizie rozwiązania przedstawiono na rys. 6.2. W prezentacji tej uwzględniono 9 pierwszych lamp drogowych wyposażonych w czujnik obecności poruszającego się samochodu (*CD_n - Car Detection*) oraz układ wykonawczy odpowiedzialny za włączanie/wyłączenie oświetlenia (*SW_n - Switch*).

Fala wyłączeń OfW (ang. the off-wave; switching off). W pierwszym eksperymencie zaprezentowano sytuację, w której samochody poruszają się w tym samym kierunku (jeden za drugim) z tą samą prędkością. Założono przy tym odległości pomiędzy samochodami większe niż 1500 metrów. Podczas tego eksperymentu spodziewano się zaobserwować efekt automatycznego wyłączania oświetlenia drogowego. Oświetlenie

[1]Można tu zastosować dowolny adaptacyjny algorytm, np. dostosowujący liczbę zapalonych lamp do prędkości samochodu.



Rys. 6.3. Cykl świecenia pojedynczej lampy.



Rys. 6.4. Korytarz zapalonych lamp po przejeździe 1 samochodu.

powinno podążać jedynie za aktualnie przejeżdżającym samochodem. Brak informacji z czujników o nadjeżdżaniu nowego samochodu skutkuje wyłączeniem zapalonych lamp [60].

Fala włączeń OnW (ang. the on-wave; switching on). Założeniem tego scenariusza badawczego jest analiza sytuacji, w której samochody poruszają się w tym samym kierunku (jeden za drugim) z tą samą prędkością. Zakładamy odległości pomiędzy samochodami mniejsze niż 440 metrów (10 x 44m, odległość słupa). Ze względu na wyjątkowo duże natężenie ruchu, spodziewano się uzyskania efektu ciągłego świecenia lamp drogowych.

6.5. Wyniki symulacji

Efekt eksperymentu z falą oświetleniową został zilustrowany na rys. 6.5, gdzie można zaobserwować koncepcję „rękawa" oświetleniowego w którym auto na drodze pierwszych 396 metrów mija 9 lamp drogowych, ustawionych co 44 metry, (lampy zapalają i gaszą się parami).

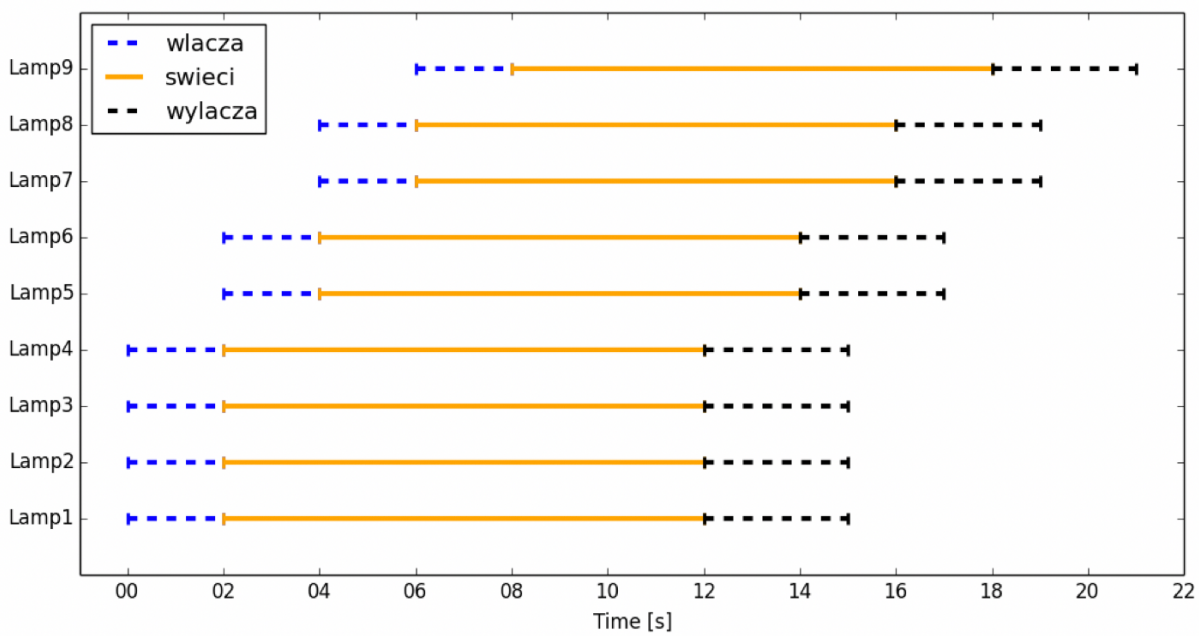
Szerokość „rękawa" oświetleniowego możliwa jest do zdefiniowania za pomocą relacji:

$$R_s = \alpha V_s \tag{6.1}$$

R_s = długość rękawa świetlnego (liczba zapalonych jednocześnie lamp) [-]
α = współczynnik ilościowy równy 0.4318 [s/m]
V_s = prędkość poruszającego się pojazdu [m/s].

Zgodnie z powyższą relacją liczba zapalonych jednocześnie lamp uzależniona jest parametrycznie od prędkości poruszającego się pojazdu. Wartość współczynnika α obliczona została na podstawie danych uzyskach podczas symulacji i reprezentuje iloraz 19 zapalonych jednocześnie lamp drogowych do prędkości poruszającego się pojazdu. Uzyskujemy w ten sposób liniową zależność zgodną z regułą, że im większa prędkość samochodu typ dłuższy „rękaw" oświetleniowy. Koordynujące urządzenie (węzeł centralny) włącza oraz wyłącza lampy w odpowiednich sekwencjach przed i za poruszającym się samochodem.

Uzyskane wyniki symulacji pozwalają stwierdzić, że po przebyciu przez samochód 836 metrów lampy, począwszy od początku drogi, przechodzą w stan całkowitego wyłączenia. Oznacza to, że pojedynczy pojazd porusza się w „rękawie" świetlnym o długości około 836 metrów (19 zapalonych lamp drogowych, $R_s = \alpha V_s = 0,4318 * 44 = 18,999$). Przed nadjeżdżającym autem zapalone są zawsze cztery lampy, natomiast pozostałych 15 lamp pali się z tyłu samochodu.



Rys. 6.5. Wykres oświetleniowej fali wyłączeń *(ang. switching off)* 9 lamp usytuowanych wzdłuż drogi na dystansie 396 [m].

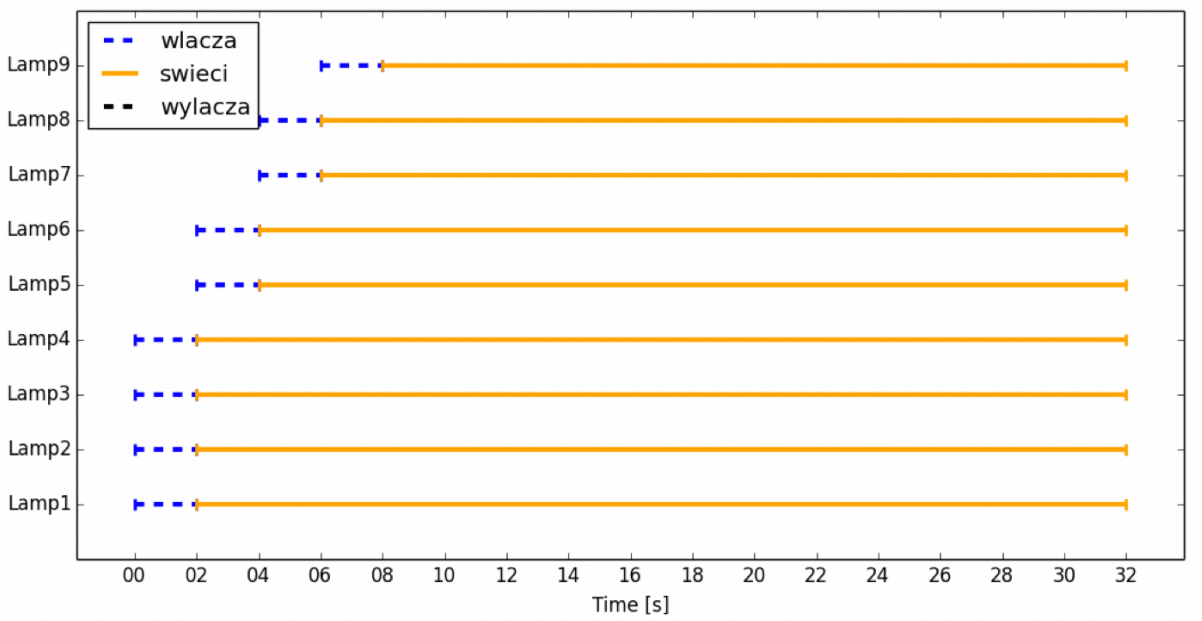
Kolejny realizowany eksperyment dotyczył oświetleniowej fali włączeń. Wyniki eksperymentu przedstawione zostały na rys. 6.6, na którym widać że, samochody poruszają się na tyle blisko siebie, że wyłączanie lamp nie następuje. Lampy świecą w trybie ciągłym czekając na informacje z czujników (o zmniejszeniu natężenia ruchu).

6.6. Wnioski i obserwacje

Podczas symulacji przeprowadzono również eksperyment na odcinku 10 000 metrów (227 lamp). Poruszające się z prędkością 44 m/s auto pokonuje tę drogę w 3 min i 47 s. Przy założeniu, że auto podróżuje w tunelu 19 sekundowym (836 m / 19 lamp drogowych), wykorzystuje się jedynie 8,37% wszystkich dostępnych lamp. Oczywiście w przypadku odcinków dłuższych niż 10 kilometrów, zysk energetyczny będzie dużo większy.
Na rys. 6.7 przedstawiono wpływ dystansu na procentową liczbę zapalonych jednocześnie lamp drogowych. Wpływ dystansu na generowane oszczędności jest oczywiście znaczący (w naturalny sposób największy progres osiąga się na dystansach do 100 km).

Na podstawie doświadczenia osiągniętego w wyniku przeprowadzonych symulacji oraz obserwacji, sformułowano następujące kryterium optymalności:

$$J = \frac{\tau_{on}}{\tau_{p1p}} \tag{6.2}$$



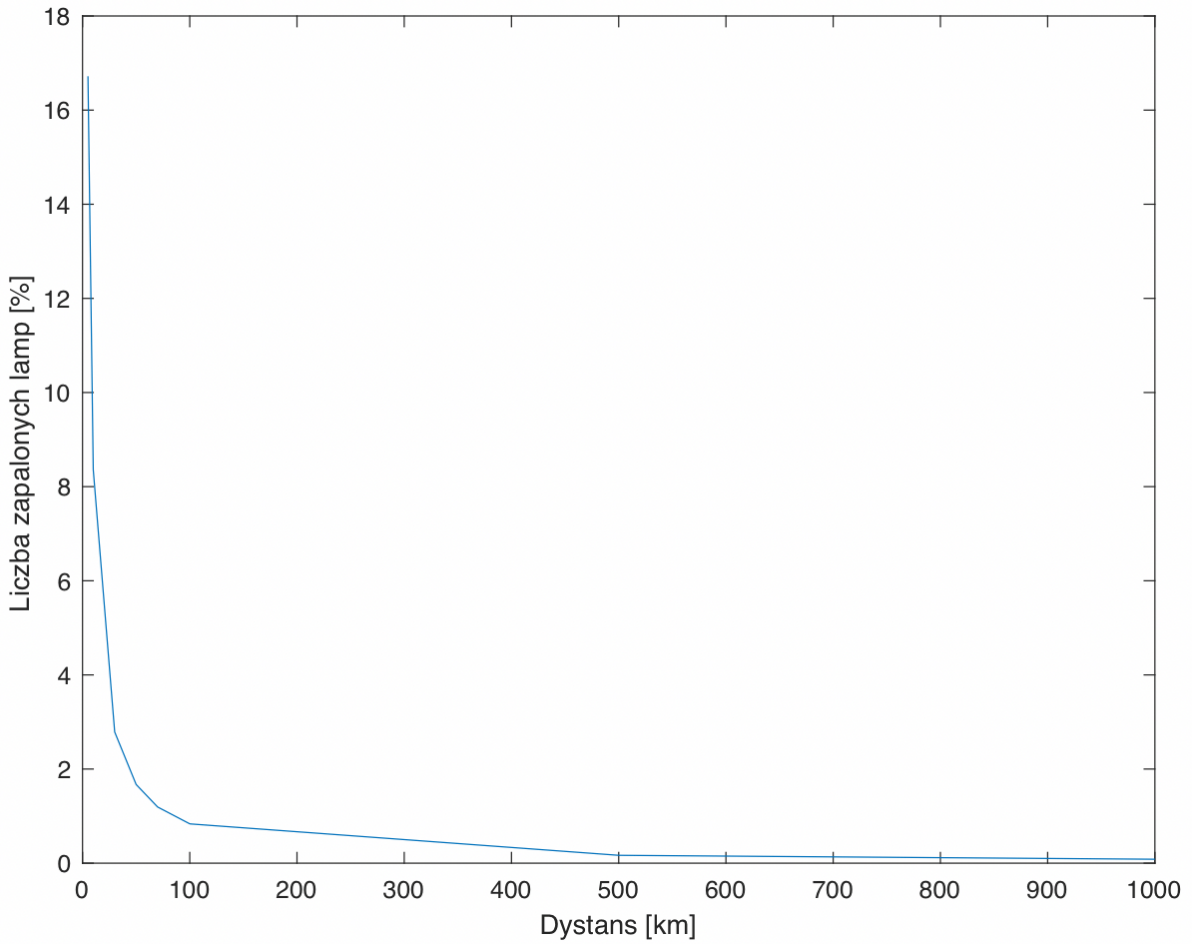
Rys. 6.6. Wykres oświetleniowej fali włączeń *(ang. switching on)* 9 lamp usytuowanych wzdłuż drogi na dystansie 396 [m].

τ_{on} = średnia liczba włączonych lamp
τ_{p1p} = średni czas przejazdu samochodu.

Działanie tego kryterium sprawdzone zostało na danych zamieszczonych w tab. 6.1, a wynikających z analizy przejazdu samochodu, poruszającego się z różnymi prędkościami na drodze 10000 metrów.
Wyniki zawarte w tab. 6.1 przedstawiono też graficznie na rys. 6.8. Jak widać, im większa prędkość poruszającego się samochodu, tym większa liczba włączonych jednocześnie lamp. Natomiast zależność liczby włączonych od czasu przejazdu ma hiperboliczny charakter.

6.7. Optymalizacja przepływu pakietów w sieci PDS

Jednym z rozpatrywanych kryteriów podczas projektowania systemów automatyki jest kryterium szybkości przesyłania pakietów pomiędzy urządzeniami pomiarowymi a systemem podejmowania decyzji. W analizowanym przypadku informacje pomiarowe przekazywane z czujników wykrywających poruszający się samochód do centralnego systemu komputerowego przekazywane są za pośrednictwem złożenia dwóch typów sieci (BACnet oraz klasycznej sieci Ethernet 1Mb/s). Wielkość przesyłanej ramki TCP przez sieć wynosi 1518 bajtów. Tak jak to przedstawiono w rozdziale 5, czas transmisji takiej liczby danych wynosi około 12 [μs]. Na przedstawionych poniżej wykresach (dla Lamp od 1 do



Rys. 6.7. Procentowy udziału liczby zapalonych lamp od całkowitej długości drogi.

5) widać jak kształtuje się w czasie ruch pakietów z danymi przenoszącymi informację pomiarową do systemu wnioskowania i analizy. Ze względu na zbliżony kształt wykresów dla Lamp 6-9 zdecydowano się na ich pominięcie. Komputerowy system wnioskowania oznaczony został na rys. 6.10 jako light-server-1. Każdy z wykresów prezentuje czas przesyłania informacji pomiarowej pochodzącej z czujników i aktuatorów, poprzez konwerter interfejsów (BACnet - TCP) aż do systemu systemu komputerowego (light-server-1).

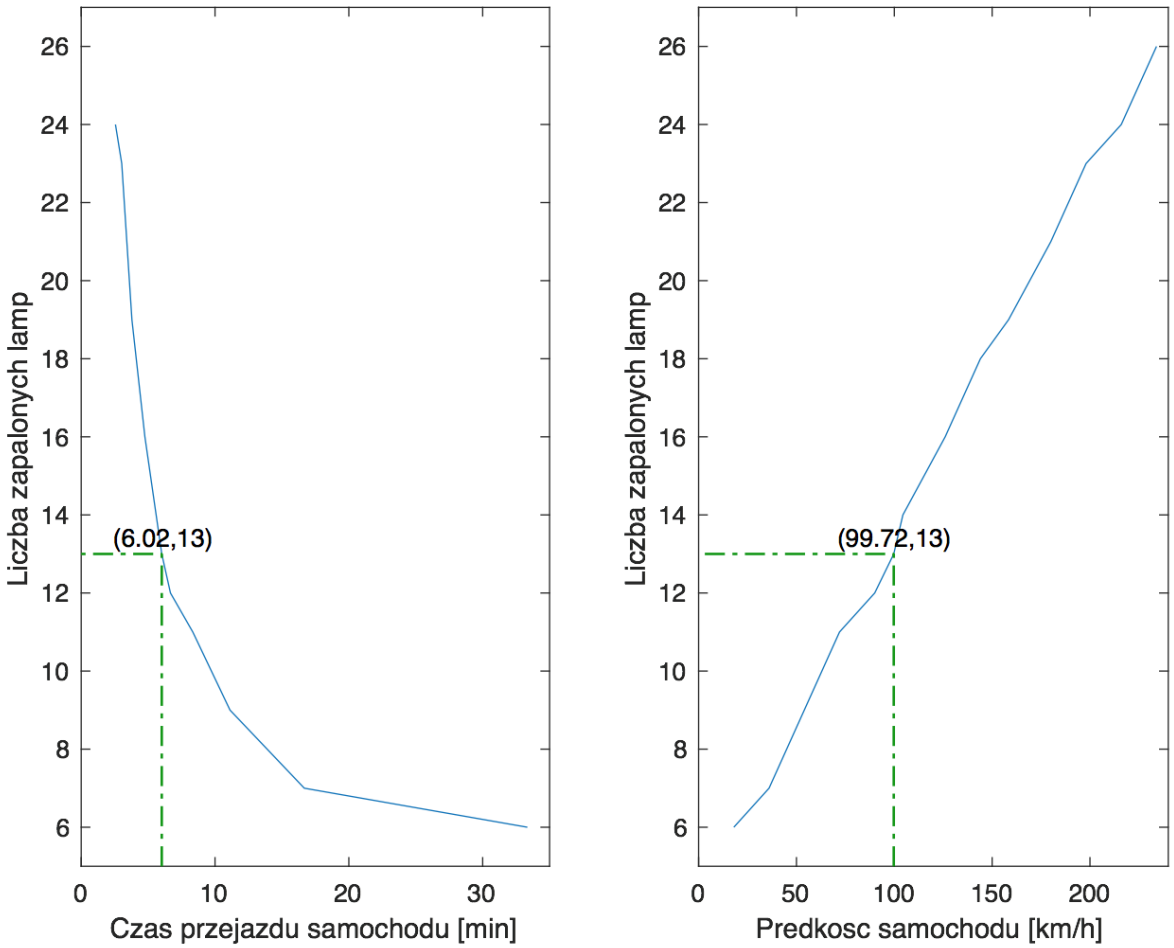
W pierwszej fazie prowadzonego eksperymentu analizowano średni czas przepływu pakietów danych pomiędzy urządzeniami pomiarowymi a pojedynczym decyzyjnym systemem komputerowym. W poddawanym weryfikacji przypadku otrzymano wartość opóźnienia pomiędzy wysłaniem pakietu z czujnika pomiarowego a systemem PDS na poziomie 0,0432 [s]. Postawiona została hipoteza, czy rekonfiguracja sieci pomiarowej poprzez rozproszenie (zduplikowanie) systemów komputerowych PDS (do wnioskowania i analizy danych) spowoduje skrócenie czasu przesyłanych pakietów z danymi. Według stawianej hipotezy w sieci pojawia się kilka lokalnych systemów komputerowych, do których

Tablica 6.1: Wyniki eksperymentu dla różnych prędkości samochodu na dystansie 10 [km].

l.p.	prędkość [km/h]	prędkość [m/s]	czas przejazdu dystansu 10000 m [min]	liczba włączonych lamp
1.	18	5	33,3333	6
2.	36	10	16,6667	7
3.	54	15	11,1111	9
4.	72	20	8,3333	11
5.	90	25	6,6667	12
6.	99,72	27,7	6,0168	13
7.	104,4	29	5,7471	14
8.	115,2	32	5,2083	15
9.	126	35	4,7619	16
10.	144	40	4,1667	18
11.	158,4	44	3,7879	19
12.	158,4	44	3,7879	19
13.	198	55	3,0303	23
14.	216	60	2,7778	24
15.	234	65	2,5641	26

podłączone są podgrupy urządzeń pomiarowo sterujących.
Sieć po rekonfiguracji ma strukturę przedstawioną na rys. 6.10. Wprowadzone zostały trzy serwery służące do analizy danych pomiarowych i generowania sygnału sterującego (lamp-server-1, lamp-server-2 oraz lamp-server-3). Każdy z serwerów działa w wyodrębnionej podsieci. W takiej konfiguracji początkowe 9 lamp podzielono na trzy niezależne podgrupy przypisane do każdego z serwerów. Powstały wydzielone autonomiczne podsieci PDS. W ramach wydzielonych segmentów możliwe jest zdecydowanie szybsze przesyłanie pakietów z danymi (mniejsza liczba urządzeń przesyłających dane jednocześnie), a tym samym podejmowanie szybszej reakcji systemu. W analizowanym przypadku uzyskany został przyrost wydajności na poziomie 40% w stosunku do inicjalnej architektury.

W trakcie prac badawczych przeprowadzono szereg eksperymentów dla różnych konfiguracji sieci pomiarowej. Badano wpływ liczby serwerów (zmniejszania segmentów sieci PDS, komunikujących się bezpośrednio z serwerami wnioskującymi) na szybkość przesyłu pakietów z danymi pomiarowymi. Zależność przedstawiająca wpływ wielkości segmentu na czas przesyłania pakietów przedstawiono na rys. 6.11. Wyniki służące do sporządzenia wykresu umieszczone zostały w tab. 6.2. Widoczna jest zależność potwierdzająca stawianą tezę, że im mniejsza liczba urządzeń pomiarowo-wykonawczych umieszczonych w segmentach sieci PDS, tym większa przepustowość danego segmentu sieci. W kontekście

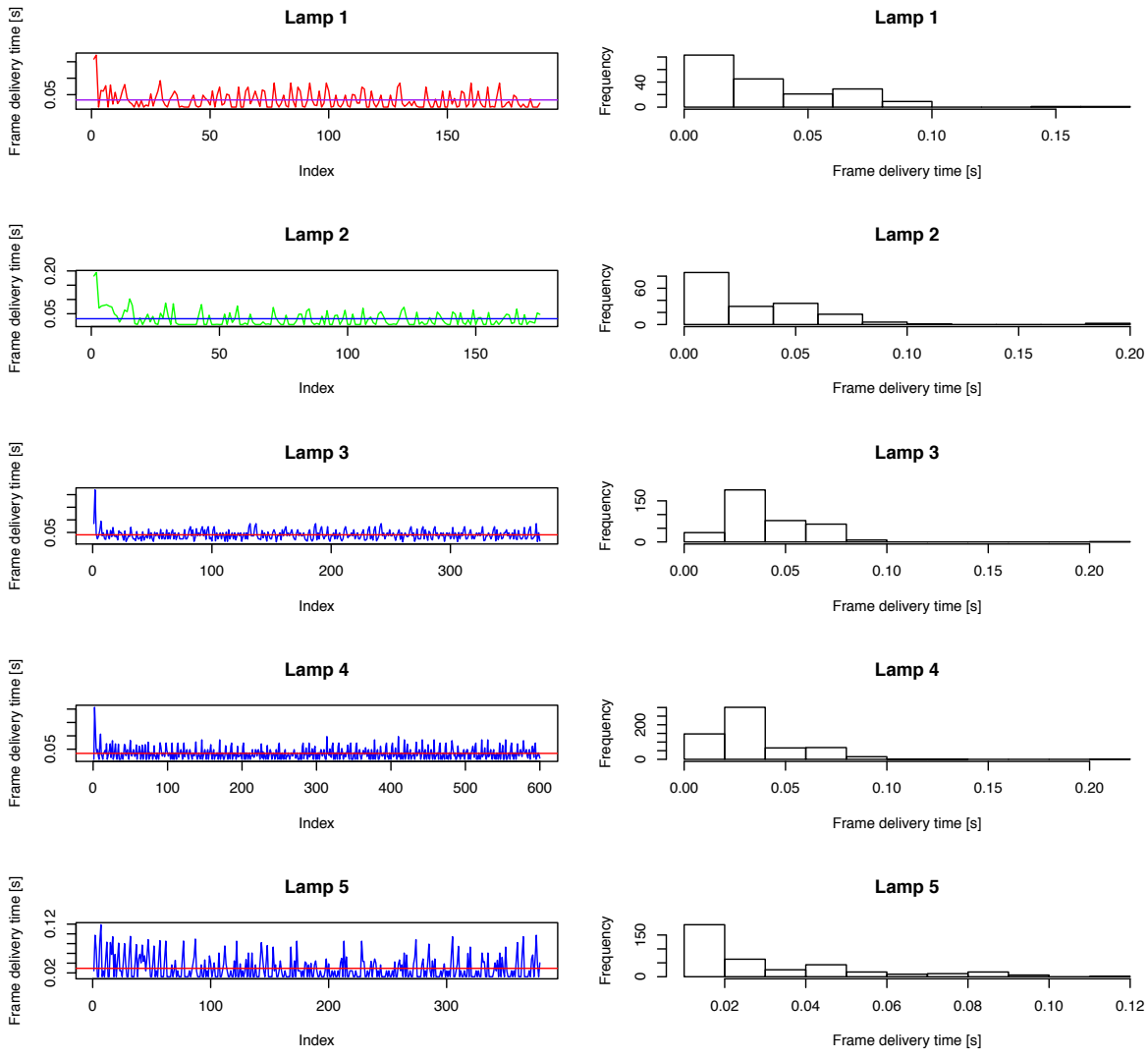


Rys. 6.8. Zależność średniej liczby zapalonych lamp w „rękawie" oświetleniowym od czasu przejazdu samochodu oraz prędkości.

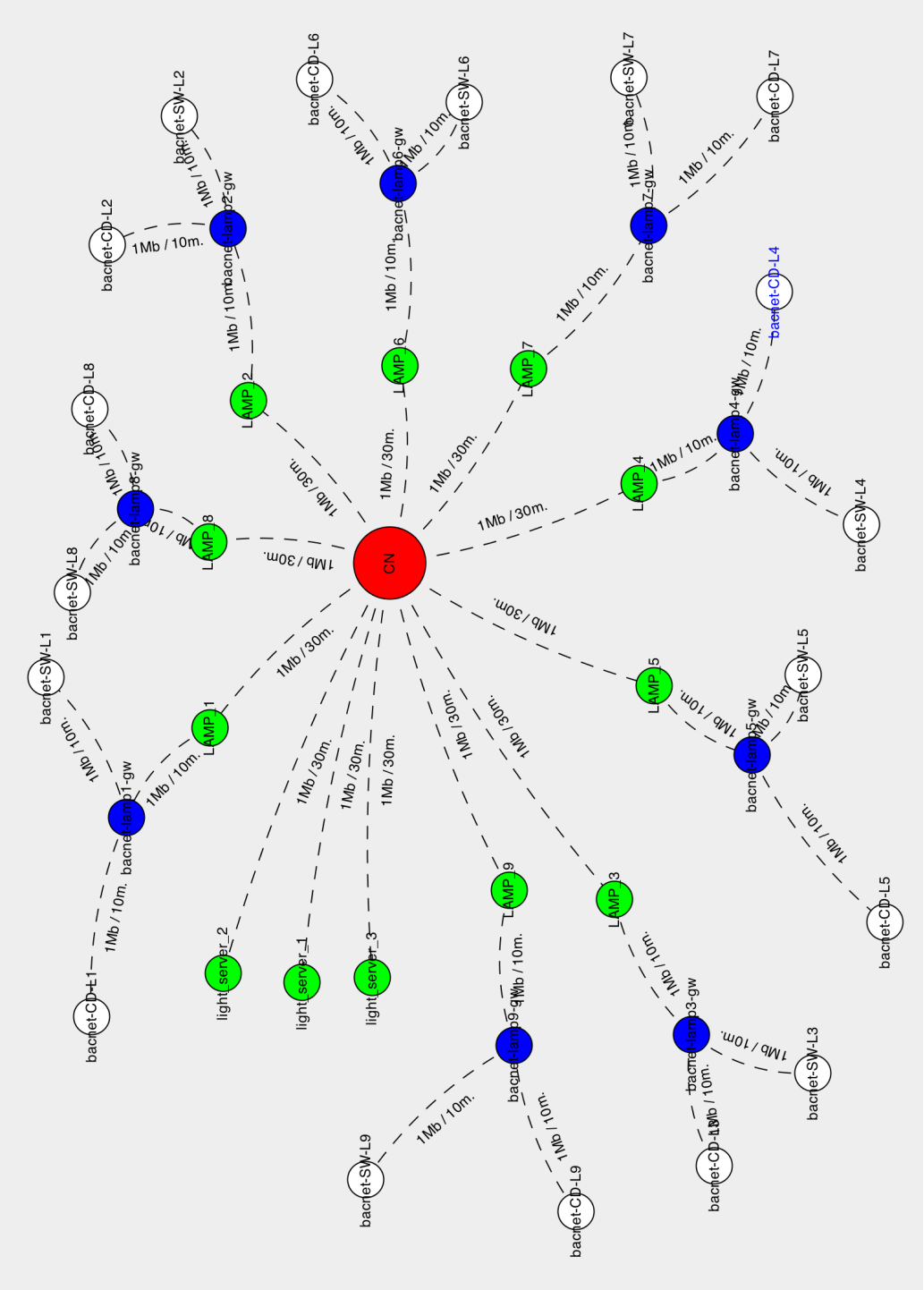
Tablica 6.2: Wyniki eksperymentu dla różnych wielkości segmentu sieci PDS.

l.p.	liczba lamp w segmencie sieci PDS	średni czas przesyłania pakietów [s]
1.	3	0.01592091
2.	4	0.01964108
3.	6	0.03075758
4.	9	0.03233172

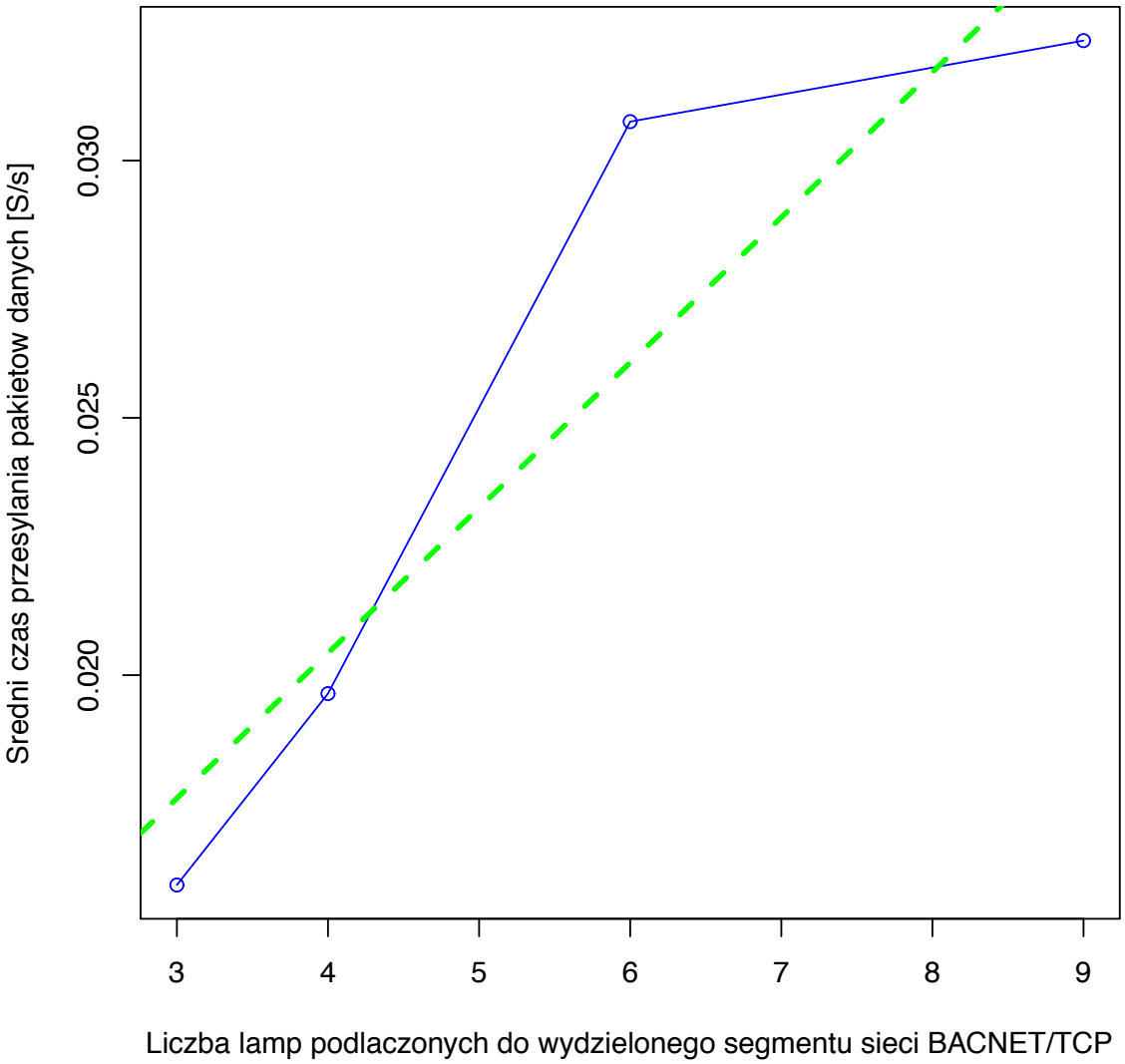
praktycznego wykorzystania powyższych wyników symulacji warto zachować kompromis dotyczący liczby wydzielonych segmentów podsieci. Najważniejszym kryterium powinien być zdefiniowany czas dostarczania danych pomiarowych z sensorów do systemów analizy i wnioskowania.



Rys. 6.9. Czas przesyłu ramki danych z informacją pomiarową dla Lamp 1 - 5.



Rys. 6.10. Sieć po rekonfiguracji z wprowadzonymi dodatkowymi serwerami do analizy danych pomiarowych.



Rys. 6.11. Wpływ wielkości segmentu podsieci PDS na czas przesyłania pakietów pomiarowych.

6.8. Podsumowanie

Zaletą opracowanej metody badawczej opartej na języku SMOL oraz zaimplementowanym środowisku symulacyjnym jest możliwość dogodnego opisu, wizualizacji oraz przeprowadzenia symulacji analizowanej sieci pomiarowo-diagnostyczno-sterującej (PDS) rozwiązującej różne problemy związane z systemami rozłożonymi. Projektant takiej sieci otrzymuje funkcjonalne narzędzie pozwalające sprawnie opisać strukturę systemu lub monitorowanego obiektu oraz związane z nim parametry, ograniczenia i relacje, oraz przeprowadzić odpowiednie symulacje.

Platforma SMOL została stworzona w celu rozwiązywania problemów charakterystycznych dla dowolnych rozłożonych systemów sieciowych. W niniejszym rozdziale zaprezentowano przykłady analizy specyficznego adaptacyjnego systemu autostradowego, który potwierdza uniwersalność platformy SMOL. Za jej pomocą przeprowadzono analizę opłacalności wprowadzania zmian w drogowym systemie oświetleniowym – potwierdzając racjonalność budowy systemów zarządzania oświetleniem dróg szybkiego ruchu.

7

ZAKOŃCZENIE

W pracy przedstawiono kompletną metodologię usprawniającą projektowanie oraz symulację rozproszonych sieci PDS wykorzystywanych powszechnie w inteligentnym budownictwie oraz przemyśle. W niniejszej dysertacji przedstawiono metodologię dotyczącą rozwiązania dwóch praktycznych problemów, tj. opracowania rozproszonego systemu nadzoru i diagnostyki SMO (Sieciowego Monitora Obiektu), będącego uniwersalnym narzędziem do monitorowania i przetwarzania danych pomiarowych oraz zaprojektowania i implementacji stowarzyszonego języka (SMOL) pozwalającego zarówno na opis struktur sieci PDS, jak i ich symulacji.

W pierwszej części dokonano przeglądu dostępnych obecnie systemów wspomagających zarządzanie, symulację i diagnostykę w obiektach inteligentnych. Opisano protokoły komunikacyjne wykorzystywane do przesyłania informacji pomiędzy czujnikami i aktuatorami a system diagnostyki i sterowania. Scharakteryzowano sieci BACnet, LonWorks, Modbus, EIB/KNX oraz LoRaWAN prezentując pewien zbiór najczęściej wykorzystywanych protokołów komunikacyjnych w rozwiązaniach z rodziny BAS (Building Automation Systems). Zaprezentowano również mechanizmy pozwalające w sposób kontrolowany i ustandaryzowany gromadzić duże zbiory danych pomiarowych oraz poprzez wykorzystanie środowisk rozproszonych wyszukiwać i poddawać je szybkiej analizie. Metody analizy dotyczyły zautomatyzowanego procesu wykrywania błędów i diagnostyki (FDD[1]).

Kolejna część pracy dotyczyła koncepcji budowy sprzętowo-programowej platformy Sieciowego Monitora Obiektu (SMO) służącej do diagnostyki i monitorowania obiektów przemysłowych. Jednym z głównych celów budowanej aplikacji było opracowanie systemu o dużym stopniu automatyzacji (samoczynności). Zaprojektowano moduł wnioskujący

[1]ang. *Fault Detection and Diagnostics*.

pozwalający na wykorzystanie baz danych SQLite oraz MySQL do przechowywania informacji o zdyskretyzowanych funkcjach przynależności, determinujących (rozmyty) sposób wykorzystywania zmiennych lingwistycznych. W oparciu o system zapytań SQL zaimplementowano bazę reguł, służącą do uzyskiwania informacji o stopniu spełnienia obranych kryteriów wyboru. Zbudowano również system wizualizacji procesu wnioskowania, pozwalający na ciągłe śledzenie poprawności działania opracowanego modułu wnioskującego.

W ramach prac projektowych nad system SMO opracowano również koncepcję diagnostycznej szyny danych (DSB - Diagnostic Service Bus) zaimplementowanej w oparciu o architekturę ESB (Enterprise Service Bus) pozwalającą na integrację wielu protokołów komunikacyjnych jednocześnie (SOAP, REST, TCP/IP, JMS, JDBC, HTTP). Szyna pozwala na podłączenie nie tylko aplikacji komputerowych, ale również umożliwia obsługę opracowanych modułów SMO. Dzięki opracowanej koncepcji moduły SMO mogą komunikować się nie tylko pomiędzy sobą w jednej lokalizacji ale również pomiędzy rozproszonymi obiektami przemysłowymi. Z punktu widzenia budowy i eksploatacji systemów pomiarowych, otrzymuje się w ten sposób rozwiązanie pozwalające na dokonywanie integracji czujników z systemami przetwarzania informacji pomiarowych. Wspólna szyna danych posiada jeden standard zapisu i przesyłania informacji. Zastosowanie architektury zorientowanej na usługi umożliwiło podłączenie dowolnych modułów wspomagających procesy diagnostyczne w inteligentnych rozwiązaniach. Podczas prac badawczych opracowano moduł FDD oraz moduł klasyfikujący dane pomiarowe z wykorzystaniem klasyfikatora Bayesa.

W dalszej części pracy dokonano przeglądu możliwości wykorzystania koncepcji języków DSL we współczesnej automatyce i informatyce stosowanej. Zaprezentowana została również szczegółowa analiza dotycząca samego języka SMOL. Opisano konstrukcje semantyczne wykorzystane do opisu struktur sieci PDS. Język SMOL wyposażony został w mechanizmy pozwalające na definiowanie urządzeń pomiarowych (SAN), węzłów transferujących (TN), expanderów (EX) oraz funkcji transformujących (TR).

Następnie przeprowadzono szczegółowe studium symulacji dyskretnej, charakteryzując sposoby i techniki symulacji procesowej, zdarzeniowej i mieszanej. Przedstawiono również opis zastosowania języka SMOL oraz opracowanego parsera do generowania kodu symulacyjnego. Podano też prosty przykład inteligentnego budynku ilustrujący metodę opisu sieci PDS w języku SMOL oraz sposób przeprowadzania symulacji w środowisku SMOL-Sim.

W dalszej części prezentowano przykład praktycznego wykorzystania języka SMOL oraz środowiska SMOL-Sim. W pierwszej części prezentowano wyniki dotyczące inteligentnego zarządzania oświetleniem autostrad (badania zlecone na potrzeby zewnętrznej firmy). W pozostałej części zaprezentowano metodę optymalizacji przepływu pakietów w rozpatrywanej sieci PDS. Pokazano możliwości rekonfiguracji sieci poprzez analizę wy-

ników symulacji otrzymanych w środowisku SMOL-Sim.

7.1. Podsumowanie rozprawy

Przedstawiona koncepcja architektury oraz implementacji rozproszonego systemu nadzoru i diagnostyki SMO pozwala na wdrażanie systemów w dowolnych obiektach mieszkalnych, biurowych oraz przemysłowych. Zaproponowano również technikę umożliwiającą wymianę informacji pomiędzy wieloma współpracującymi ze sobą modułami SMO z wykorzystaniem szyny danych DSB (zgodnie z architekturą (SOA) zorientowaną serwisowo) - *dowodząc w ten sposób pierwszej tezy pomocniczej rozprawy.* Opracowany został również język modelowania i symulacji SMOL wyrażający wszystkie niezbędne funkcje systemu SMO - *potwierdzając przez to drugą tezę pomocniczą.*

Podsumowany oryginaly wkład rozprawy to:

- opracowanie koncepcji rozproszonych modułów SMO i wykorzystanie ich do diagnostyki budynkowej (prace [59, 61])
- projekt inteligentnego systemu umożliwiającego monitorowanie parametrów rozproszonych obiektów przemysłowych po przez sieć (prace [55, 56])
- diagnostyka obiektów przemysłowych z wykorzystaniem wspólnej magistrali usług DSB zrealizowana zgodnie z architekturą zorientowaną serwisowo - SOA (praca [57])
- zdefinowanie koncepcji, semantyki języka SMOL oraz środowiska SMOL-Sim (prace [59, 58])
- praktyczne zastosowanie środowiska symulacyjnego do analizy i optymalizacji systemu inteligentnego oświetlenia autostrad (praca [60]).

7.2. Kierunki dalszych badań

Możliwym kierunkiem dalszych badań jest rozszerzenie zaprezentowanego rozwiązania, tak aby moduły SMO mogły komunikować się na większe odległości wymieniając informacje pomiarowo-diagnostyczne. W kontekście opracowanej architektury sprzętowej warte rozważenia jest zwielokrotnienie możliwych opcji komunikacyjnych, poza inicjalne zaimplementowanym wsparciem dla technologii ZigBee. Warte rozważenie wydaje się również zaproponowanie alternatywnego źródła zasilania pozwalającego na używanie modułów jako jednostek całkowicie autonomicznych. Zasadne wydaje się również wspieranie

większej liczby protokołów komunikacyjnych obecnych w automatyce przemysłowej i budynkowej np. BacNET, MQTT itd.

Warto też aby dalsze prace rozwojowe objęły stowarzyszone środowisko wizualizacyjne dla platformy SMOL-Sim. Obecnie trwają prace nad rozszerzeniem semantyki dodatkowych instrukcji pozwalających dogodniej zarządzać kodem programu (mniejsze fragmentu sieci będą opisywane w odrębnych plikach załączanych do projektu głównego). Ponadto realizowane są prace nad rozszerzeniem symulatora o nowy moduł uwzględniający potrzeby symulacji urządzeń komunikujących się w standardzie MQTT.

Dodatki

A

SYMULACJA SIECI PDS

```
tn "fdd-server", {
    ip "2"
}

tn "eth1", {
    ip "1"
}

tn "eth3", {
    ip "3"
}

tn "eth4", {
    ip "4"
}


expander "bacnet-floor1-gw", {
    connect "eth1"
    model "bacnet"
}

expander "bacnet-floor2-gw", {
    connect "eth3"
    model "bacnet"
}

expander "bacnet-floor3-gw", {
    connect "eth4"
    model "bacnet"
}
```

```
/* FLOOR 3 */
// HVAC controller
san "hvac-bacnet-floor3",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "500"
}

// Light sensor
san "light-bacnet-floor3",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-floor3-1",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-floor3-2",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-floor3-3",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "1500"
}
```

```
san "temp-sensor-bacnet-floor3-4",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-floor3-5",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "600"
}

san "temp-sensor-bacnet-floor3-6",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "400"
}

san "temp-sensor-bacnet-floor3-7",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "400"
}

san "temp-sensor-bacnet-floor3-8",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "400"
}

san "temp-sensor-bacnet-floor3-9",{
    connect "bacnet-floor3-gw"
    destAddress "2"
    freq "400"
}

/* FLOOR 2 */
// HVAC controller
san "hvac-bacnet-floor2",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "500"
}

// Light sensor
san "light-bacnet-floor2",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-floor2-1",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-floor2-2",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-floor2-3",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-floor2-4",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-floor2-5",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "550"
}

san "temp-sensor-bacnet-floor2-6",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "550"
}

san "temp-sensor-bacnet-floor2-7",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "560"
}

san "temp-sensor-bacnet-floor2-8",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "570"
}

san "temp-sensor-bacnet-floor2-8",{
    connect "bacnet-floor2-gw"
    destAddress "2"
    freq "570"
}

/* FLOOR 1 */
// HVAC controller
san "hvac-bacnet-floor1",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}
```

```
// Light sensor
san "light-bacnet-floor2",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-1",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-2",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-3",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "1500"
}

san "temp-sensor-bacnet-4",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-5",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-6",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-7",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-8",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-9",{
    connect "bacnet-floor1-gw"
    destAddress "2"
    freq "500"
}


action "draw", {
    fullmap "true"
}

// stop after 60 sek
action "sim", {
    stop "60000"
}
```

B

OPIS SYMULACYJNY SIECI PDS

```
// Transfer node definition
// Define a network adapter
// Assign IP address
tn "eth1", {
    ip "1"
}

tn "eth3", {
    ip "3"
}

tn "server", {
    ip "2"
}

tn "eth4", {
    ip "4"
}

expander "bacnet-gw-1", {
    connect "eth4"
    model "bacnet"
}

// Expander node definition
// Define CAN network gateway
// Connect to the Transfer node
expander "can-gw", {
    connect "eth1"
    model "bacnet"
}
```

```
// Expander node definition
// Define Bacnet network gateway
// Connect to the Transfer node
expander "bacnet-gw", {
    connect "eth3"
    model "bacnet"
}

// Temp sensor definition
// Connect sensor to CAN bus
// Set measure values final destination address (2)
san "temp-sensor-can-1",{
    connect "can-gw"
    destAddress "2"
    freq "2000" }

san "temp-sensor-can-2",{
    connect "can-gw"
    destAddress "2"
    freq "200"
}

san "temp-sensor-can-3",{
    connect "can-gw"
    destAddress "2"
    freq "200"
}

san "temp-sensor-can-4",{
    connect "can-gw"
    destAddress "2"
    freq "200"
}
```

```
san "temp-sensor-bacnet-1",{
    connect "bacnet-gw"
    destAddress "2"
    freq "200"
}

san "temp-sensor-bacnet-2", {
    connect "bacnet-gw"
    destAddress "2"
    freq "200"
}

san "temp-sensor-bacnet-3", {
    connect "bacnet-gw"
    destAddress "2"
    freq "56"
}

san "temp-sensor-bacnet-31", {
    connect "bacnet-gw-1"
    destAddress "2"
    freq "500" }

san "temp-sensor-bacnet-32", {
    connect "bacnet-gw-1"
    destAddress "2"
    freq "500"
}

san "temp-sensor-bacnet-33", {
    connect "bacnet-gw-1"
    destAddress "2"
    freq "500"
}

// Draw graph
action "draw", {
    fullmap "true"
}

// Run simulation phase. Stop simulation after 60 sec.
action "sim", {
    stop "60000"
}
```

C

OPIS SYMULACYJNY SIECI ZARZĄDZAJĄCEJ OŚWIETLENIEM AUTOSTRAD

```
tn "light_server", {
    ip "1"
}

tn "LAMP_1", {
    ip "2"
}

tn "LAMP_2", {
    ip "3"
}

tn "LAMP_3", {
    ip "4"
}

tn "LAMP_4", {
    ip "5"
}

tn "LAMP_5", {
    ip "6"
}

tn "LAMP_6", {
    ip "7"
}

tn "LAMP_7", {
    ip "8"
}

tn "LAMP_8", {
    ip "9"
}

tn "LAMP_9", {
    ip "10"
}

expander "bacnet-lamp1-gw", {
    connect "LAMP_1"
    model "bacnet"
}

expander "bacnet-lamp2-gw", {
    connect "LAMP_2"
    model "bacnet"
}

expander "bacnet-lamp3-gw", {
    connect "LAMP_3"
    model "bacnet"
}

expander "bacnet-lamp4-gw", {
    connect "LAMP_4"
    model "bacnet"
}

expander "bacnet-lamp5-gw", {
    connect "LAMP_5"
    model "bacnet" }
```

```
expander "bacnet-lamp6-gw", {
    connect "LAMP_6"
    model "bacnet"
}

expander "bacnet-lamp7-gw", {
    connect "LAMP_7"
    model "bacnet"
}

expander "bacnet-lamp8-gw", {
    connect "LAMP_8"
    model "bacnet"
}

expander "bacnet-lamp9-gw", {
    connect "LAMP_9"
    model "bacnet"
}

// Lamp 1 car detector
san "bacnet-CD-L1",{
    connect "bacnet-lamp1-gw"
    destAddress "1"
    freq "500"
}

// Lamp 1 switch
san "bacnet-SW-L1",{
    connect "bacnet-lamp1-gw"
    destAddress "1"
    freq "500"
}

// Lamp 2 car detector
san "bacnet-CD-L2",{
    connect "bacnet-lamp2-gw"
    destAddress "1"
    freq "500"
}

// Lamp 2 switch
san "bacnet-SW-L2",{
    connect "bacnet-lamp2-gw"
    destAddress "1"
    freq "500"
}

// Lamp 3 car detector
san "bacnet-CD-L3",{
    connect "bacnet-lamp3-gw"
    destAddress "1"
    freq "500"
}

// Lamp 3 switch
san "bacnet-SW-L3",{
    connect "bacnet-lamp3-gw"
    destAddress "1"
    freq "500"
}

// Lamp 4 switch
san "bacnet-SW-L4",{
    connect "bacnet-lamp4-gw"
    destAddress "1"
    freq "500"
}

// Lamp 4 car detector
san "bacnet-CD-L4",{
    connect "bacnet-lamp4-gw"
    destAddress "1"
    freq "500"
}

// Lamp 5 switch
san "bacnet-SW-L5",{
    connect "bacnet-lamp5-gw"
    destAddress "1"
    freq "500"
}

// Lamp 5 car detector
san "bacnet-CD-L5",{
    connect "bacnet-lamp5-gw"
    destAddress "1"
    freq "500"
}

// Lamp 6 switch
san "bacnet-SW-L6",{
    connect "bacnet-lamp6-gw"
    destAddress "1"
    freq "500"
}

// Lamp 6 car detector
san "bacnet-CD-L6",{
    connect "bacnet-lamp6-gw"
    destAddress "1"
    freq "500"
}

// Lamp 7 switch
san "bacnet-SW-L7",{
    connect "bacnet-lamp7-gw"
    destAddress "1"
    freq "500"
}
```

```
// Lamp 7 car detector
san "bacnet-CD-L7",{
    connect "bacnet-lamp7-gw"
    destAddress "1"
    freq "500"
}

san "bacnet-SW-L8",{
    connect "bacnet-lamp8-gw"
    destAddress "1"
    freq "500"
}

san "bacnet-CD-L8",{
    connect "bacnet-lamp8-gw"
    destAddress "1"
    freq "500"
}

san "bacnet-SW-L9",{
    connect "bacnet-lamp9-gw"
    destAddress "1"
    freq "500"
}

san "bacnet-CD-L9",{
    connect "bacnet-lamp9-gw"
    destAddress "1"
    freq "500"
}

action "draw", {
    fullmap "true"
}

// stop after 5 sek
action "sim", {
    stop "5000"
}
```

SPIS OZNACZEŃ SYMBOLI I SKRÓTÓW

Symbol lub skrót	Opis	strona def.
PDS	Pomiarowo Diagnostyczno Sterujący	58
SMOL	Language for Networked Systems for Monitoring Objects	64
CN	Węzeł centralny (ang. Central Node)	67
TN	Węzeł przenoszący (ang. Transfer Node)	68
EX	Expander	69
TR	Transformator	69
SAN	Sensor/Actuator Node	70
SMO	Sieciowy Monitor Obiektu	43

SPIS RYSUNKÓW

SPIS TABLIC

BIBLIOGRAFIA

[1] Abramczyk A.: Sterowanie systemami HVAC. *Inteligentny Budynek*, no. 2, 2016.

[2] Abramczyk A.: Sterowanie oświetleniem. *Inteligentny Budynek*, no. 1, 2017.

[3] Arentowicz K.: Zastosowanie standardu Zigbee do zdalnego sterowania urządzeniami pomiarowymi. *Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej, Seria: Technologie Informacyjne*, Gdańsk 2006.

[4] Ahuja A., Moore R.: Integracja automatyki budynkowej oznacza lepszą wydajność. *Inteligentny Budynek*, no. 3, 2016.

[5] Alexandru M., Fiasche M., Pinna C., Taisch M., Fasanotti L., Grasseni P.: Building a Smart Maintenance Architecture using Smart Devices: A web 2.0 based Approach. *IEEE 2nd Internation Forum on Research and Technologies for Security and Industry Leveranging a better tomorrow (RTSI)*, 2016.

[6] Biernacki A., Brachman A., Chróst Ł., Domańska J., Głomb P., Grochla K., Nowak M., Nowak S., Pecka P.: Symulacja Sieci Komputerowych. *ITIS PAN*, 2009.

[7] Bryndza L.: LPC2000. Mikrokontrolery z rdzeniem ARM7. *BTC*, Warszawa 2007.

[8] Hutchison B., Schmidt M.T., Wolfson D., Stockton M.: SOA programming model for implementing Web services, Part 4, An introduction to the IBM Enterprise Service Bus. *https://www.ibm.com/developerworks/library/ws-soa-progmodel4/*, 2017.

[9] Bobcow A., Dąbkowski M.: Biometryczna kontrola dostępu. *Wydawnictwo PAK*, no.4, 2007.

[10] Borowiec W., Górski W.: Integracja usług z wykorzystaniem architektury ESB. *Praca Magisterska*, AGH Krakow 2008.

[11] Chapman T.: NDL 1.0 – Network design language. *SCTE Cable-Tec Expo*, 2008.

[12] Chappell D.: ESB – Magistrala usług korporacyjnych. *Helion*, 2014.

[13] Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, vol. 51, issue 1, January 2008.

[14] Dearle F.: Groovy for domain-specific languages – Second Edition. *PACKT*, ISBN 978-1-8496-9540-4, 2015.

[15] Demianiuk S.: Skanery laserowe w kontroli dostępu. *Napędy i Sterowanie*, no. 6, 2006.

[16] Doliński J.: Mikrokontrolery AVR w praktyce. *BTC*, Warszawa 2003.

[17] Domschke W., Drexel A.: Introduction to Operations Research, *Springer*, Berlin, 4th edition, 1998.

[18] Du Z., Fan B., Jin X., Chi J.: Fault detection and diagnosis for buildings and HVAC systems using combined neural networks and subtractive clustering analysis. *Building and Environment*, vol. 73, pp. 1-11, 2014.

[19] Du Z., Jin X., Yang Y.: Fault diagnosis for temperature, flow rate and pressure sensors in VAV systems using wavelet neural network. *Applied Energy*, vol. 86, pp. 1624-1631, 2009.

[20] DSL: A theoretical survay (https://www.researchgate.net/publication/228670370_Domain_specific_languages_a_theoretical_survey)

[21] Dutta J., Roy S.: IoT-Fog-Cloud based architecture for Smart City. *Prototype of a Smart Building*, IEEE ISBN 978-1-5090-3519-9, 2017.

[22] Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. *Prentice Hall PTR Upper Saddle River*, ISBN 0131858580, NJ, USA 2009.

[23] Fan C., Xiao F., Yan C.: A framework for knowledge discovery in massive building automation data and its application in building diagnostics. *Automation in Construction*, vol. 50, pp. 81-90, 2015.

[24] Fisher D.: BACnet and LonWorks: A White Paper. *PolarSoft Inc*, 1996.

[25] Fulmański P., Grzanek M.: Logika rozmyta i SQL. *Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej, Seria: Technologie Informacyjne*, Gdańsk 2006.

[26] Gabrysiak R.: Kontrola dostępu – przykładowe rozwiązania. *Elektroinstalator*, no.10, 2012.

[27] Ghosh D.: DSLs in Action. *Manning*, ISBN 1935182455, 1st Edition, 2011.

[28] http: *www.automatyka-budynkowa.com - artykuły branżowe - integracja systemów SSWin i CCTV*. http://www.automatyka-budynkowa.com/artykuly/240/integracja-systemow-sswin-i-cctv.htm, (2018).

[29] http: *siemens.com*. http://www.buildingtechnologies.siemens.com/bt/global/en/products/hvac-products/home-automation-system/synco%E2%84%A2-living/pages/rrv934.aspx, (2017).

[30] http: *European Commission*. http://cordis.europa.eu/result/rcn/190985_pl.html, (2017).

[31] http: *EksperBudowlany.pl - jak wybrać domofon*. http://www.ekspertbudowlany.pl/artykul/id2520,jak-wybrac-domofon, (2018).

[32] http: *Kamery cctv, przemysłowe - Akcesoria do wideodomofonów i domofonów - Eura-Tech*. http://www.eura-tech.eu/akcesoria-do-wideodomofonow-i-domofonow,2,16, (2018).

[33] http: *Polak zaprojektował najlepszy dom świata 2017 roku według magazynu Wallpaper*. http://www.fpiec.pl/post/2017/01/15/polakzaprojektowalnajlepszydomswiata, (2018).

[34] http: *Inteleko - Dom inteligenty i Ekonomiczny - dom inteligentny, domy inteligentne, dom ekologiczny, domy ekologiczne*. http://www.inteleko.pl/pl/pinteligentne-budynki-uzytecznosci-publicznej_97.html, (2018).

[35] http: *Inteligentne Instalacje*. http://www.inteligentne-instalacje.pl/pl/48,dzwiekowe-systemy-ostrzegawcze-i-rozglaszania-dso-dsr.html, (2017).

[36] http: *RFT1000 Czytnik linii papilarnych*. http://www.roger.pl/pl/kontrola-dostepu/oferta-uzupelniajaca-do-systemow-racs-5-i-racs-4/biometryczne/rft1000-czytnik-linii-papilarnych.html, (2017).

[37] http: *EnOcean produkty*. http://www.wago.pl/rozwiazania/technika-budynkowa/automatyka-budynkowa/opis-systemu/enocean/, (2017).

[38] http: *Łatwa automatyzacja dzięki LonWorks*. http://www.wago.pl/rozwiazania/technika-budynkowa/automatyka-budynkowa/opis-systemu/lonworks/, (2018).

[39] http: *nblog : Język dziedzinowy w praktyce (Domain-Specific Language Tools)*. http://zine.net.pl/blogs/nuwanda/archive/2007/01/23/dsl-tools.aspx, (2018).

[40] http: *Cloud computing - Wikipedia*. https://en.wikipedia.org/wiki/Cloud_computing, (2018).

[41] http: *Digital Addressable Lighting Interface - Wikipedia*. https://en.wikipedia.org/wiki/Digital_-Addressable_Lighting_Interface, (2018).

[42] http: *Budynek zeroenergetyczny – Wikipedia*. https://pl.wikipedia.org/wiki/Budynek_zeroenergetyczny, (2018).

[43] http: *About ANSI*. https://www.ansi.org/about_ansi/overview/overview, (2018).

[44] http: *Products — TM Technologie*. http://en.tmtechnologie.pl/products, (2018).

[45] http: *Overview SimPy 3.0.11 documentation*. https://simpy.readthedocs.io/en/, (2017).

[46] http: *Inteligentne budynki. Materiały wykładowe - Mariusz Nowak, Instytut Informatyki, Politechnika Poznańska*. http://www.cs.put.poznan.pl/mnowak/IB/IB-2.pdf, (2017).

[47] http: *System kontroli dostępu - szeroki zakres funkcji, stała opieka serwisowa*. http://www.cs.pl/systemy/system-kontroli-dostepu/, (2017).

[48] http: *BACnet System Architecture*. http://www.cylon.com/us/assets/media/Pdfs/Resources/BACnet-Solution-NA.pdf, (2017).

[49] Katipamula S., Brambley M.: Methods for Fault Detection, Diagnostics, and Prognostics for Building Systems — A Review, Part I. *Journal of HVAC&R*, vol. 11, no. 1, January 2005.

[50] Koperski B., Nowak M., Szymborska A.: Wykorzystanie standardu LoRaWAN do budowy bezprzewodowych sieci sensorowych w inteligentnych budynkach. *Napędy i Sterowanie*, Czerwiec 2016.

[51] Kowalczuk Z., Dembek J., Sołtyka W.: *Charakterystyka Energetyczna Budynków* [Z. Kowalczuk (Ed.)], PWNT Pomorskie Wydawnictwo Naukowo-Techniczne, ISBN 978-83-926806-1-1, pp. 1-4, Gdańsk 2010.

[52] Kowalczuk Z., Wszołek J.: Sieciowy monitor obiektu wspierający prace eksperta. *Inżynieria Wiedzy i Systemy Ekspertowe* [A. Grzech, K. Juszczyszyn, H. Kwasnicka, N.T. Nguyen (Eds.)], ISBN 978-83-60434-55-0, AOW EXIT, cz. III (Projektowanie i realizacja systemów ekspertowych), pp. 329-337, Warszawa 2009.

[53] Kowalczuk Z.: Mathematical Modeling and Simulation. *Lecture Materials*, Department of Robotics and Decision Systems, Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technology, 2018.

[54] Kowalczuk Z., Wszołek J.: System monitorowania obiektów poprzez sieć. *Zastosowanie Technologii Informatycznych w Zarządzaniu Wiedzą*. Pomorskie Wydawnictwo Naukowo-Techniczne PWNT, pp.77-83, 2009.

[55] Kowalczuk, Z., Wszołek, J.: Monitoring objects over networks. *Systems Science*, vol. 35, no. 3, pp. 49-53, 2009.

[56] Kowalczuk, Z., Wszołek, J.: Sieciowy monitoring i diagnostyka obiektów. *Systemy Wykrywające i Tolerujące Usterki*, pp.227-234, 2009.

[57] Kowalczuk Z., Wszołek J.: Diagnostyka obiektów przemysłowych z wykorzystaniem wspólnej magistrali usług Diagnostic Service Bus (DSB). *Problemy Eksploatacji*, vol. 2, pp.121-130, 2011.

[58] Kowalczuk Z., Wszołek J.: Modelowanie struktur sieci diagnostyczno-pomiarowych z wykorzystaniem języka SMOL. *Aktualne Problemy Automatyki i Robotyki*, Wrocław: Politechnika Wrocławska, pp.696-705, 2014.

[59] Kowalczuk Z., Wszołek J.: Networked Object Monitor – A distributed system for monitoring, diagnostics and control of complex industrial facilities. *Metrology and Measurement Systems*, vol. 19, no. 3, pp. 521-530, 2012.

[60] Kowalczuk Z., Wszołek J.: Analysis of economical lighting of highways in the environment of SMOL language. *Metrology and Measurement Systems*, 2017.

[61] Kowalczuk Z., Wszołek J.: Network monitoring and diagnostics of buildings. – Z. Kowalczuk (Ed.): *Detecting, Analysing and Fault-Tolerant Systems*, pp. 227-234, PWNT, Gdańsk, 2009.

[62] Krzaczek M., Kowalczuk Z.: Thermal Barrier as a technique of indirect heating and cooling for residential buildings. *Energy and Buildings*, vol. 43, no. 4, pp. 823-837, 2011.

[63] Krzaczek M., Kowalczuk Z.: Gain Scheduling control applied to Thermal Barrier in systems of indirect passive heating and cooling of buildings. *Control Engineering Practice*, vol. 20, no. 12, pp. 1325-1336, 2012.

[64] Kubski P., Kowalczuk Z.: Wymiana ciepła – podstawy teoretyczne. *Charakterystyka Energetyczna Budynków* [Z. Kowalczuk (Ed.)], PWNT Pomorskie Wydawnictwo Naukowo-Techniczne, ISBN 978-83-926806-1-1, pp. 219-226, Gdańsk 2010.

[65] Kubski P., Kowalczuk Z.: Podstawy oceny energetycznej systemów ogrzewania, chłodzenia i zaopatrzenia w ciepłą wodę użytkową. *Charakterystyka Energetyczna Budynków* [Z. Kowalczuk (Ed.)], ISBN 978-83-926806-1-1, PWNT Pomorskie Wydawnictwo Naukowo-Techniczne, pp. 227-236, Gdańsk 2010.

[66] Krysiak K.: Sieci komputerowe. Kompedium. Wydanie II., *Helion*, ISBN 978-83-246-3540-5, 2005.

[67] Kwasnowski P., Ożadowicz A.: Systemy Inteligentnych Budynków. *Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej*, materiały wykładowe, 2017.

[68] LonMark Open Systems Deliver End-User Benefits, http://www.lonmark.com

[69] LoRa Alliance Website, LoRa Technology, https://www.lora-al-liance.org/What-Is-LoRa/Technology

[70] Marsic I.: Computer Networks. Performance and Quality of Service. *Rutgers, The State University of New Jersey*, New Jersey, 2013.

[71] Mohamed N., Lazarova-Malnar S., Al-Jaroodi J.: SaSBDaaS: Smart Building Diagnostics as a Service on the Cloud. *IEEE*, ISBN 978-1-4673-8473-5, 2016.

[72] Myeong-in C., Keonhee C., Jun Y., Lee P., Kyu Hee J., Sunghwan P., Sehyun P.: Design and implementation of IoT-based HVAC system for future zero energy building. *The First International Workshop on Smart Edge Computing and Networking*, ISBN: 978-1-5090-4339-2, 2017.

[73] Namburu S. M., Azam M. S., Luo J., Choi K., Pattipati K. R.: Data-driven modeling, fault diagnosis and optimal sensor selection for HVAC chillers. *IEEE Transactions on Automation Science and Engineering*, vol. 4, pp. 469-473, 2007.

[74] Nawrocki W.: Rozproszone systemy pomiarowe. *Wydawnictwo Komunikacji i Łączności*, Warszawa, 2006.

[75] Nielson, H.R., Nielson, F.: Semantics with applications: a formal introduction. *John Wiley& Sons, Inc.*, New York, NY, USA, 1992.

[76] Newman H.M.: BACnet - The New Standard Protocol. *Electrical Contractor*, wrzesien 1997.

[77] Nordmann A.: A Survey on Domain-Specific Languages in Robotics. *SIMPAR 2014: Simulation, Modeling, and Programming for Autonomous Robots*, pp. 195-206, 2014.

[78] Nowak M.: Zabezpieczenia przeciwpożarowe w inteligentnych budynkach. *Inteligentny Budynek*, no.12, 2011.

[79] Nowak M., Nowak M.: Rozproszony system SCADA do sterowania i zarządzania instalacjami w inteligentnych budynkach. *Napędy i Sterowania*, no.12, 2012.

[80] OBrien W., Bourdoukan P., Delisle V., Yip S.: Net ZEB design processes and tools. *Modeling, Design, and Optimization of Net-Zero Energy Buildings*, 2015.

[81] ONeill Z., Pang X., Shashanka M., Haves P., Bailey T.: Model-based real-time whole building energy performance monitoring and diagnostics. *Journal of Building Performance Simulation*, vol. 7, pp. 83-99, 2014.

[82] Ożadowicz A.: Analiza porównawcza dwóch systemów sterowania inteligentnym budynkiem - systemu europejskiego EIB/KNX oraz standardu amerykańskiego na bazie technologii LonWorks. *Rozprawa Doktorska*, Kraków 2006.

[83] Petykiewicz P.: Nowoczesna instalacja elektryczna w inteligentnym budynku. *Centralny Ośrodek Szkolenia i Wydawnictw SEP*, Warszawa 2001.

[84] Page B., Kreutzer W.: The Java Simulation Handbook - Simulating Discrete Event Systems with UML and Java. *Shaker Verlag*, ISBN 3-8322-3771-2, Germany 2005.

[85] Pramod J., Folwer M.: *NoSQL Kompendium Wiedzy*, Helion, 2015.

[86] Przekwas P., Grzybowski A.: Aplikacja monitorująca trajektorie ruchu na podstawie danych z urządzenia mobilnego. *Projekt Dyplomowy Inżynierski*, Gdańsk 2016.

[87] Seip G.: Czas magistrali. *Inteligentny Dom* no.1, Wrocław 2000.

[88] Seip G., Hagedorn K.: Project Engineering for EIB Installations 4th revised edition. *European Installation Bus Association (EIBA)*, 1998.

[89] SEMTECH, LoRa, Wireless RF Solutions,
https://www.semtech.com/images/mediacenter/collateral/ism_sg.pdf

[90] SEMTECH, SX1272/73-860 MHz to 1020 MHz Low Power Long Range Transceiver, Datasheet, http://www.semtech.com/images/datasheet/sx1272.pdf

[91] Sołtyka W., Kowalczuk Z. Energia słoneczna – źródło energii odnawialnej – niedostatecznie wykorzystane w Polsce. *Konsulting Polski*, ISSN 2353-5091, no. 3/4, pp. 45-47, 2014.

[92] Skuba M.: Inteligentne struktury sieciowe w technologii LON. *Rozprawa Doktorska*, Kraków 2011.

[93] Swan B.: Internetworking with BACnet. A first look at networking in Bagnet. *Alerton Technologies, Inc*, 1997.

[94] Szepietowski M.: Budynek inteligentny - opis rozwiązań. *Elektroinfo*, no.5, Warszawa 2002.

[95] Szymczyk I. Otwarte i zamkniete systemy zarzadzania budynkiem inteligentnym. *I Ogolnopolskie Seminarium Studentow i Doktorantow*, AGH Krakow 2005.

[96] Tacklim L., Seonki J., Dongjun K., Lee Won P., Sehyun P.: Design and Implementation of Intelligent HVAC System Based on IoT and Bigdata Platform. *2017 IEEE International Conference on Consumer Electronics*, 2017.

[97] Tadeusiewicz R.: Inteligentny budynek-laboratorium jako element formujący wieź między badaniami naukowymi a wdrożeniem wyników do praktycznych realizacji. *Napędy i Sterowanie*, no.12, 2015.

[98] Tepic S., Pejic P., Domsic J., Milhaldinec H., Dzapo H.: IBMS - Intelligent Building Management System Framework. *MIPRO*, 2015.

[99] White T.: Hadoop Kompletny przewodnik. Analiza I przechowywania danych. *Helion*, ISBN 978-83-283-1457-3, 2016.

[100] Zaharia M., Chowdhury M., Das T., Ankur D., Ma J., McCauley M., Franklin M., Shenker S., Stoica I.: Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing.*NSDIÍ12 Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, 2012.

[101] Zaharia M., Chowdhury M., Franklin M., Shenker S., Stoica I.: Spark: Cluster Computing with Working Sets. *HotCloudÍ10 Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 2010.

[102] Żabicki D.: Sterowanie oświetleniem i oświetleniem ewakuacyjnym w inteligentnym budynku. *ElektroInfo*, no.9, 2015.