mgr inż. **Marcin Michał MIROŃCZUK**
Politechnika Białostocka
Wydział Elektryczny

SYSTEMY ZARZĄDZANIA BAZĄ DANYCH I ARCHITEKTURA AGENTOWA W SŁUŻBACH RATOWNICZYCH PAŃSTWOWEJ STRAŻY POŻARNEJ

Database management system and agent architecture into fire service

Praca naukowa współfinansowana ze środków Europejskiego Funduszu Społecznego, środków Budżetu Państwa oraz ze Środków Budżetu Województwa Podlaskiego w ramach projektu "Podlaska Strategia Innowacji - budowa systemu wdrażania

Streszczenie

W artykule przedstawiono autorską klasyfikację Systemów Zarządzania Bazą Danych (SZBD) oraz opisano możliwość zastosowania architektury agentowej w służbach ratowniczych Państwowej Straży Pożarnej PSP. W pierwszej części artykułu dokonano autorskiej klasyfikacji, przeglądu i opisu SZBD już od pewnego czasu dobrze znanych i opisanych np. rozwiązania katalogowe, relacyjne jak i aktualnie rozwijających się np. rozwiązania obiektowe, koncepcyjne czy też oparte o rozszerzony język znaczników (*ang. extensible markup language – XML*). Dotychczas zastosowanie samych baz danych i SZBD w służbach ratowniczych PSP jest mocno ograniczone. Zazwyczaj ich użycie sprowadza się do ewidencji i rejestracji zdarzeń z ewentualnym minimalnym ich wsparciem od strony informacyjnej dla Kierującego Działaniami Ratowniczymi KDR. Przedstawienie więc przekrojowej analizy SZBD daje możliwość szerszego spojrzenia na ewentualne zastosowania niektórych rozwiązań w służbach ratowniczych, w szczególności tych mających na celu wspieranie akcji ratowniczo-gaśniczych. W drugiej części artykułu skupiono się na właśnie takim rozwiązaniu. Przedstawiono w nim zarys systemu opartego o architekturę agentową. Opisano podstawowe funkcje oraz sposób działania systemu opartego o taką architekturę. Na końcu dokonano podsumowania z wyszczególnieniem technik i powstających problemów projektowych przy realizacji omawianej platformy.

Summary

This article describes a new approaches and classification of Database Management System DBMS. In this article also describe potential using of agent architecture into fire service. The first part of this article is a review of DBMS. In this section describes DBMS historically oldest solutions like a hierarchical database model, relational database model and describes the newest solutions like a object oriented database model, conceptual model or model base on extensible markup language – XML. Actually applying of DBMS in polish fire service is strongly limited. Usually their use moves to keep a record of fire service events. Eventually DBMS are used to the minimum support of decision-makers. The proposed review gives the wider glance on uses DBMS in rescue services. In the second part author describes a outline decision support system for fire service which based on agent data base management system. This section describes a system's functions and his way of working. The end the article consist of summary where author describes techniques and problems appears in project cycle of agent based platform.

Słowa kluczowe: systemy zarządzania bazą danych, SZBD, architektura agentowa, programowanie agentowe, projektowanie agentowe, przegląd systemów zarządzania bazą danych

Keywords: Data base management system, agent architecture, agent programming, agent architecture, review of data base management system

1. Wstęp

W obecnych Systemach Zarządzania Bazą Danych (SZBD) termin „dane" może być definiowany, w zależności od stopnia złożoności, rodzaju i typu składowanych w nich treści, na trzy sposoby [1-3]:

Definicja 1. Szczątkowe, nieuporządkowane sygnały, które mogą pochodzić ze źródeł pierwotnych albo wtórnych, tworzonych wewnątrz jak i na zewnątrz organizacji

Definicja 2. Synonim słowa „informacja" określający w ten sposób rezultat integrowania i porządkowania danych, które w ten sposób nabierają znaczenia

Definicja 3. Wyrażenie równoznaczne dla wiedzy – informacji wartościowej i zaakceptowanej, integrującej dane, fakty i informacje. Wiedza może być dostępna oraz ukryta. Wiedza dostępna jest wiedzą spisaną, skodyfikowaną, ogólnie dostępną przez wszystkich, którzy są nią zainteresowani. Wiedza ukryta jest natomiast wiedzą indywidualną, specyficzną, znaną tylko jej posiadaczowi (decydentowi), trudną do sformalizowania.

Traktowanie danych w SZBD według pierwszej definicji nie ma praktycznego znaczenia, ze względu na to, iż w systemie przechowujemy z reguły już w pewien sposób uporządkowaną informację. Definicja 2. odzwierciedla dzisiejsze (czas w jakim powstało niniejsze opracowanie) podejścia do danych zebranych w różnego rodzaju SZBD. Definicja 3. nawiązuje natomiast do aktualnie budowanych i badanych Systemów Zarządzania Wiedzą. W dalszej części tego tekstu, pod nazwą SZBD będą rozumiane systemy zarządzania informacją jak i wiedzą. Aktualnie, jeśli SZBD rozpatrywane są w kontekście zarządzania wiedzą, można spotkać gamę różnorodnych terminów, które na poziomie semantycznym są synonimami. Do grupy tych terminów należą: System Zarządzania Bazą Informacji SZBI lub System Zarządzania Informacją SZI oraz System Zarządzania Bazą Wiedzy SZBW lub System Zarządzania Wiedzą SZW.

Na potrzeby badań nad zastosowaniem (SZBD) w służbach ratowniczych Państwowej Straży Pożarnej (PSP) budowany był system o roboczej nazwie FINE (Fine it's not EWID), którego celem miało być uzupełnienie o potrzebne funkcje oprogramowania dostarczanego przez firmę Abakus o nazwie EWID 9x (EWID99 lub EWID 93) [4, 5]. Kontynuację i rozszerzenie tych badań stanowić ma platforma Hybrydowego Inteligentnego Systemu Wspomagania Decyzji (HSWD), oparta o katalogową lub obiektową, rozproszoną bazę danych. HSWD koncepcyjnie stanowi System Zarządzania Bazą Wiedzy o architekturze rozproszonej z wnioskowaniem na podstawie przypadków CBR [6-9]. Równolegle prowadzone są także badania nad przechowywaniem i pozyskiwaniem, potrzebnych atrybutów do przestrzennej bazy przeciwpożarowej GIS [10]. Wyniki z tych badań mogą zostać wykorzystane do budowy systemu oraz wnieść dodatkową wiedzę na temat reprezentowania i tworzenia formalnych raportów w planowanym systemie, które można by było w łatwy i efektywny sposób przetwarzać (wyszukiwać, zapisywać) w wybranym typie – rozproszonym, zcentralizowanym – zarządzania bazą danych [6, 10-12]. Rozpatrywane jest także wykorzystanie, obok katalogowej bazy, natywnej bazy XML, jako bazy wiedzy, w której będzie można przechowywać i wydobywać opis ontologiczny poszczególnych przypadków. Rozwiązanie to zostało zaproponowane ze względu na to, iż baza natywna XML może przechowywać bogaty opis semantyczny zdarzeń i akcji ratowniczo-gaśniczych, w których biorą udział siły i środki służb ratowniczych PSP [8, 13].

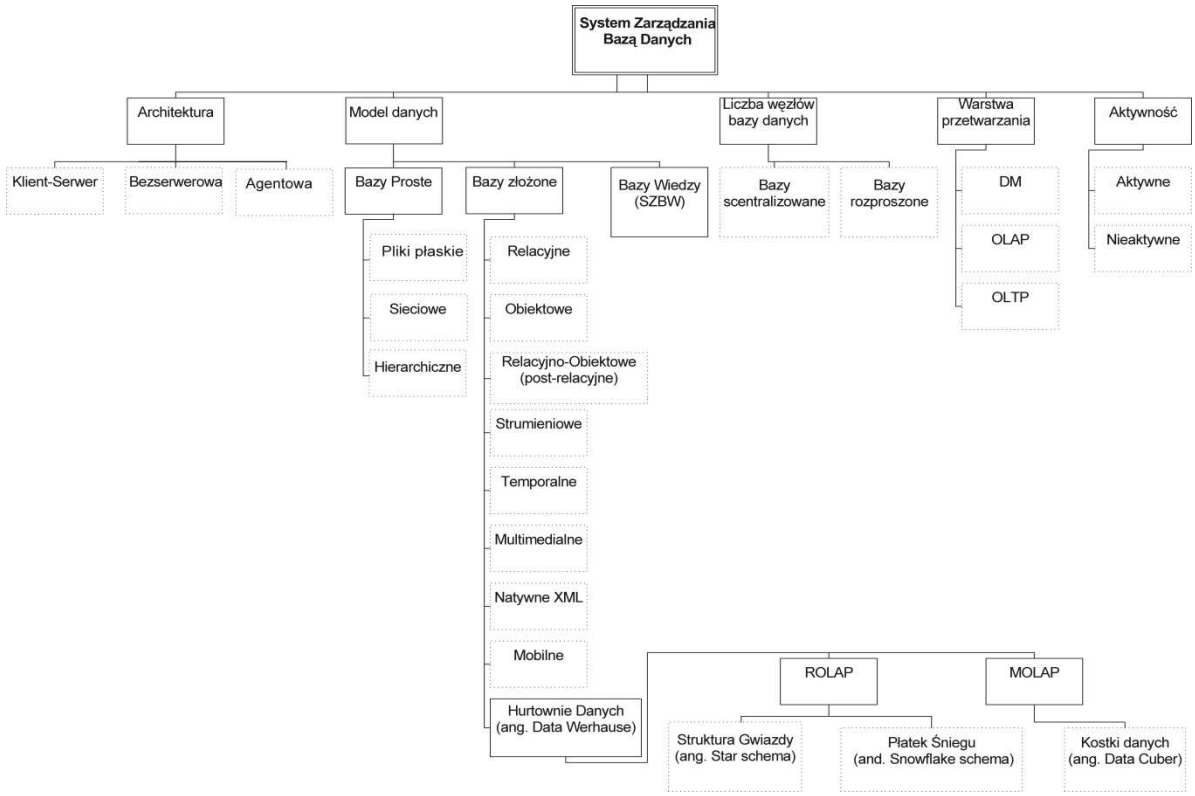
Jak widać w zespole badawczym operuje się dużą ilością terminologii i pojęć z zakresu SZBD. W związku z tym, w celu zachowania przejrzystości komunikacji i zrozumienia pomiędzy członkami zespołu badawczego, powstała idea utworzenia formalnego słownika pojęć, którymi można operować w kontekście SZBD. W wyniku jego

tworzenia, powstała prezentowana w artykule klasyfikacja SZBD wraz z omówieniem konkretnego zastosowania polegającego na wykorzystaniu architektury agentowej.

W pierwszej części artykułu (rozdział 2) autor przedstawił klasyfikację i przegląd SZBD ze szczególnym naciskiem na opis nowych trendów, pojęć i rozwiązań w dziedzinie baz danych. W drugiej części artykułu (rozdział 3) autor opisał zarys możliwości wykorzystania architektury agentowej do rozwiązywania problemów: decyzyjnych pojawiających się podczas akcji ratowniczo-gaśniczych oraz analitycznych.

2. Klasyfikacja i przegląd Systemów Zarządzania Bazą Danych

Klasyfikacja i zarazem przegląd opisowy SZBD powstał na podstawie analizy dostępnej dla autora literatury opisującej zagadnienia związane z ogólnie pojętymi SZBD [11, 14-19]. Klasyfikację tę przedstawia Rycina 1.



Ryc. 1 Klasyfikacja współczesnych SZBD

Źródło: Opracowanie własne

Ryc. 1. przedstawia diagram autorskiego podziału współczesnych SZBD w zależności od ich: architektury (podrozdział 2.1), modelu danych (podrozdział 2.2), liczby węzłów (podrozdział 2.3), rodzaju warstwy przetwarzania danych (podrozdział 2.4) i aktywności

(podrozdział 2.5). Poszczególne gałęzie i liście ww. kategorii zostały rozwinięte i omówione w dalszych sekcjach niniejszego artykułu.

2.1. Architektura Systemów Zarządzania Bazą Danych

Architektura ZSBD rozumiana jest jako specyfikacja procesu komunikacji, który zawiera przetwarzanie i przesyłanie danych, informacji oraz wiedzy w obrębie systemu informacyjnego, pomiędzy jego poszczególnymi jednostkami składowymi [20]. Do architektury SZBD zaliczamy rozwiązania komunikacji typu: klient-serwer (podrozdział 2.1.1), bezserwerowa (podrozdział 2.1.2) oraz agentowa (podrozdział 2.1.3).

2.1.1. Architektura klient-serwer

Architektura klient-serwer (*ang. client-server*) jest to architektura aplikacji, w której przetwarzanie jest podzielone między maszynami (procesami) działającymi jako klienci i maszynami działającymi jako serwery np. aplikacji lub baz danych [15]. Klienci pozyskują zasoby informacji poprzez proces komunikacji z serwerem, na którym są one składowane. Klient ma możliwość pozyskiwania informacji z jednego (architektura scentralizowana - podpunkt 2.3.1) lub wielu serwerów jednocześnie (architektura rozproszona - podpunkt 2.3.2). Architektura oprogramowania klient-serwer rozdziela funkcjonalność oraz odciąża system, tym samym zwiększając jego elastyczność. Ułatwia ona przy tym wprowadzanie zmian w różne części tego systemu. Nie ma wymagań formalnych odnośnie lokalizacji obu procesów, teoretycznie mogą one znajdować się na jednym komputerze (baza lokalna). W praktyce jednak serwer umieszczany jest na innym komputerze niż procesy klienta, a komunikacja odbywa się poprzez sieć (model dwuwarstwowy przetwarzania danych). Możliwa jest również sytuacja, w której serwer baz danych udostępnia dane klientom bezpośrednio lub pośrednio przez inny serwer pośredniczący (np. serwer WWW lub aplikacji). Wówczas architektura taka nazywana jest architekturą klient-serwer trzy lub wielowarstwową (wielowymiarową).

Główną rolą klienta jest: dostarczenie graficznego lub tekstowego interfejsu użytkownika do komunikacji z bazą danych, akceptowanie i kontrola syntaktyki zgłoszeń użytkownika do systemu, generowanie zapytań do bazy danych i transmisja ich do serwera oraz odbieranie odpowiedzi od systemu bazy danych. Natomiast do głównych ról serwera należą: akceptowanie i realizacja żądań klienta do bazy danych, kontrola autoryzacji, kontrola więzów integralności, wykonywanie zapytań i transmisja odpowiedzi do klienta, obsługa

katalogu systemowego, sterowanie pracą współbieżną oraz sterowanie odtwarzaniem stanu bazy danych.

Do zalet architektury rozproszonej (*ang. distributed system*) klient-serwer można zaliczyć m.in. to, że [6, 12]: umożliwia szeroki dostęp do istniejących baz danych, poprawia współczynnik przetwarzania (chociażby przez fakt rezydowania na różnych komputerach serwera i klienta), umożliwia pracę równoległą aplikacji, co w pewnym zakresie redukuje koszty sprzętu (maszyna, na której posadowiony jest serwer, musi być komputerem o dobrych parametrach, w odniesieniu do stacji klienckich wymagania nie są aż tak wysokie), zmniejsza koszty komunikacji w sieci (część przetwarzania odbywa się po stronie klienta, serwer przesyła z bazy tylko dane wynikowe), zwiększa spójności i bezpieczeństwa bazy danych (serwer kontroluje więzy integralności, a więc są one obowiązujące dla wszystkich pracujących aplikacji). Za wadę uznaje się [6]: złożoność i skomplikowanie oprogramowania, zależność od jakości i architektury sieci, mniejsze bezpieczeństwo i odporność na błędy.

2.1.2. Architektura bezserwerowa

Architektura bezserwerowa występuje wówczas, kiedy stosowanie procesu serwera nie jest konieczne. Istnieją bazy danych, które nie muszą być współdzielone przez wielu użytkowników w tym samym czasie. W tym celu używane są bezserwerowe bazy danych, które instalowane są na maszynie klienta i do których tylko on ma dostęp w ramach używanych przez niego aplikacji.

2.1.3 Architektura agentowa

Architekturę agentową, w kontekście systemów zarządzania bazami danych, można traktować jako część architektury typu klient-serwer. Niemniej została ona przydzielona do osobnej gałęzi klasyfikacji z tego względu, że w stosunku do architektury i rozwiązań typu klient-serwer jest w dalszym ciągu, od jej powstania w 1994 roku mało rozpropagowana i implementowana. Rozwiązań technologicznych opartych na agentach i metodologii tworzenia oprogramowania agentowego nie łączy się aktualnie bezpośrednio z bazami danych i systemami ich zarządzania. Do takiego połączenie i wyodrębnienia gałęzi powinno jednak dojść ponieważ agenci czerpią dane do swego działania z SZBD poprzez komunikację z warstwą aplikacji do ich obsługi. Zachodzi więc proces komunikacji z innym komputerem, który jest jednym z wyróżników podejścia agentowego [21]. Agent powoływany jest do życia po to, aby dostarczyć informacji lub wiedzy klientowi, który go wywołał, na zadany przez niego temat. Tak więc jednym z elementów agenta jest zdolność do komunikowania się

z istniejącymi systemami obsługującymi źródła danych czy też wiedzy. Połączenie elementu dotyczącego tego, iż agent może współpracować z warstwą aplikacji obsługującą SZBD oraz może sam przenosić się miedzy systemami, stał się czynnikiem dostatecznym do wydzielenia nowego liścia w opisywanej klasyfikacji. Dodatkowo w architekturze agentowej, aplikacja klienta czy też serwer aplikacji klienta nie musi podtrzymywać połączenia ze źródłem danych i oczekiwać zwrotu informacji. Może ona natomiast odpytywać czy agent wrócił z poszukiwaną informacją lub też zostać powiadomiona o jego powrocie.

Do tej pory nie powstała jedna zwarta definicja agenta, z tego względu poniżej zostały przytoczone dwie definicje najlepiej pasujące do kontekstu SZBD [21]:

Definicja 1. Jednostki programowe podejmujące działania w imieniu użytkownika lub innych programów, w pewnym stopniu niezależne lub autonomiczne, które działając stosują pewną wiedzę lub reprezentację celów.

Definicja 2. Zamknięty system komputerowy znajdujący się w pewnym otoczeniu, posiadający umiejętność elastycznego działania w tymże otoczeniu, polegającego na wypełnianiu celów dla jakich został stworzony.

Wspólną cechą definicji jest to, iż agent stosuje pewną swoją wiedzę o tym, jak rozwiązać zadany problem i gdzie szukać ewentualnych danych. Podczas swego działania może on kontaktować się z innymi agentami i systemami. Z tego względu agentom, jak i systemom agentowym, przypisuje się pewne cechy, np. [22]: proaktywność (*ang. proactivity*), kooperatywność (*ang. cooperavity*) czy też adaptywność (*ang. adapivity*). Natomiast w zależności od zastosowań agentów można wyróżnić ich trzy grupy [21]: agenci do zarządzania i przetwarzania informacji stosowani np. w przetwarzaniu obrazów [23, 24], negocjacji cen [25], grach [26] etc.; agenci w systemach rozproszonych np. różnego rodzaju boty internetowe; agenci modelujący systemy złożone. W zastosowaniach najczęściej mamy do czynienie z systemami mieszanymi, nie dającymi się w łatwy i przejrzysty sposób zakwalifikować do jednej z trzech wymienionych grup.

2.2. Model danych Systemów Zarządzania Bazą Danych

Przez model danych rozumiana jest architektura danych oraz zintegrowany zbiór wymagań w odniesieniu do danych [15]. Jako model danych (a w odniesieniu do konkretnej realizacji mówi się często o architekturze systemu baz danych) można też rozumieć zbiór ogólnych zasad posługiwania się danymi. W latach sześćdziesiątych i siedemdziesiątych XX wieku ujrzało światło dzienne wiele pomysłów dotyczących rozwiązania problemu zarządzania powtarzającymi się danymi. W wyniku różnych eksperymentów powstało kilka

modeli systemów baz danych. Sam problem magazynowania danych i szybkiego dostępu do nich istniał już znacznie wcześniej, jednakże rozwiązywany był przy pomocy bardzo prymitywnych pomysłów i języków programowania [14]. Model danych składa się z takich podkategorii modeli, jak: prosty model danych (podrozdział 2.2.1), złożony model danych (podrozdział 2.2.2) oraz model silnie semantyczny utożsamiany i stosowany w systemach zarządzania bazą wiedzy (podrozdział 2.2.3).

2.2.1. Prosty model danych

W prostym modelu danych, dane reprezentowane są za pomocą rekordów zgrupowanych w strukturach plików. W tym przypadku dostępnymi operacjami na rekordach są: odczyt i zapis rekordu [15]. Do prostego modelu danych zaliczane są: model pliku płaskiego, model hierarchiczny oraz model sieciowy, które zostały omówione poniżej.

Model pliku płaskiego

Model pliku płaskiego, nazywany inaczej modelem kartotekowym (kartotekowe bazy danych, *ang. flat file database*), ,model ten stanowi najprostszą organizację bazy danych. Informacje w tym modelu przechowywane są w jednym płaskim pliku lub ich zbiorze – kartotece. Pliki nie są połączone ze sobą w żaden sposób, a ich interpretacja dokonywana jest poprzez zewnętrzną aplikację [11].

Model hierarchiczny

Model hierarchiczny przypomina obecny schemat zapisu systemu plików. Zakłada, że związki pomiędzy danymi mogą zachodzić tylko typu jeden do wielu (1:N) tj. rekord typu ojciec może posiadać wiele powiązań z rekordami typu syn [11]. Do operowania i przetwarzania danych w tym modelu używa się funkcji wbudowanych w bazę danych, napisanych w wybranym języku programowania (w tak zwanym języku gospodarza). Integralność danych jest przestrzegana poprzez kontrolę następujących więzów: rekordu podrzędnego nie można wstawić dopóki nie zostanie powiązany on z rekordem nadrzędnym, skasowanie rekordu nadrzędnego powoduje usunięcie wszystkich powiązanych z nim rekordów podrzędnych (dzieci), jeżeli podrzędny typ rekordu ma związane dwa lub więcej nadrzędnych typów rekordów, to rekord podrzędny musi zostać powielony dla każdego rekordu podrzędnego [15].

Model sieciowy

Model sieciowy, w przeciwieństwie do modelu hierarchicznego, zakładał, że mogą istnieć związki między danymi typu wiele do wielu (N:N) [11]. Pod względem historycznym sieciowy model danych jest uważany za następcę modelu hierarchicznego. Do operowania

i przetwarzania danych w tym modelu, tak jak w modelu hierarchicznym, używa się funkcji napisanych w wybranym języku programowania. Ważne jest przy tym, aby pamiętać, że język programowania gospodarza i system bazy danych to dwa oddzielne systemy oprogramowania, które łączy się poprzez wspólny interfejs programowania aplikacji (*ang. application programming interface – API*). Integralność w modelu sieciowym dotyczy głównie określenia członkostwa w kolekcji i trybu wstawiania [15].

2.2.2. Złożony model baz danych

W przeciwieństwie do prostych modeli danych, modele złożone najczęściej traktują dane w sposób zdyscyplinowany poprzez użycie ścisłych narzędzi matematycznych np. teorii zbiorów do posługiwania się danymi w przypadku modelu relacyjnego. Ponadto istnieje duża niezależność pomiędzy programami operującymi na danych, a danymi składowanych w bazie danych. Dane w tym modelu mogą przybierać też inny niż tekstowy charakter np. binarny w przypadku baz multimedialnych lub obiektów w przypadku baz obiektowych etc. Do złożonego modelu danych zaliczane są: model relacyjny, model obiektowy, model strumieniowy, model relacyjno-obiektowy, model temporalny, model multimedialny, model natywny, model mobilny oraz model hurtowni danych. Wszystkie ww. modele zostały opisane poniżej.

Model relacyjny

Model relacyjny został opracowany przez E. F. Codda w latach 70-80 [27] i od mniej więcej połowy lat osiemdziesiątych stał się podstawą architektury większości popularnych systemów baz danych. Podstawową strukturą danych w tym modelu jest relacja. W bazach danych relacja przedstawiana jest w postaci tabeli. Relacja jest zbiorem rekordów, z których każdy ma taką samą strukturę, lecz różne wartości. Każdy rekord odpowiada jednemu wierszowi tablicy. Każdy wiersz (rekord) musi posiadać co najmniej jeden atrybut, odpowiadający jednej kolumnie tablicy. Cały model relacyjny bazuje na dwóch gałęziach matematyki – teorii mnogości (zbiorów) oraz rachunku predykatów pierwszego rzędu [28]. Wykorzystanie formalnej matematycznej struktury relacyjnej wpłynęło na sukces relacyjnych baz i pozwoliło m.in. na [11]: przeprowadzanie automatycznego sprawdzania konstrukcji, zagwarantowało wykonalność operacji i spójność zbiorów danych oraz, najważniejsze, – dało możliwość zadawania dowolnych zapytań wyrażanych w strukturalnym języku zapytań (*ang. structured query language – SQL*) [29] przez użytkowników, a nie tylko zdefiniowanych przez projektantów systemów opartych o bazy danych.

Relacyjny model danych jest wydajniejszy od wcześniej omawianych prostych modeli np. plików tekstowych. W przeciwieństwie do modeli prostych, w bazach relacyjnych pola są określonego typu, dlatego można w nich łatwo przechowywać teksty, liczby, daty czy też wartości binarne Dzięki wykorzystaniu języka SQL obsługa relacyjnych baz danych stała się prosta i wszechstronna. Dzięki językowi SQL w relacyjnych bazach danych kolejność rekordów przestała mieć jakiekolwiek znaczenia, ponieważ język SQL umożliwił nie tylko wybranie rekordów, aktualizacje i wstawianie, ale również ich sortowanie (funkcja manipulacji danymi, *ang. data manipulation language – DML*). SQL oprócz manipulowania danymi umożliwia także wprowadzanie zmiany danych w bazie i jej strukturze (funkcja opisu danych, *ang. data definition language – DDL*).

Systemy bazodanowe do zarządzania danymi opartymi o model relacyjny określane są jako – Relacyjny System Zarządzania Bazą Danych RSZBD (*ang. relational database managament system – RDBMS*).

Model obiektowy

Termin obiektowy w dzisiejszej informatyce ma wiele różnych znaczeń. Po raz pierwszy został on zastosowany przez twórcę paradygmatu obiektowego Norwega Kristena Nygarrda i odnosił się do pierwszego obiektowego języka programowania Simula, a potem do ich całej grupy np. Smalltak, Java, C++. Od niedawna terminem tym operuje się w dziedzinie baz danych. Zasadniczą różnicą między obiektowymi językami programowania a obiektowymi bazami danych jest to, że te pierwsze nie wymagają składowania i operacji na trwałych obiektach tj. obiekty programowe istnieją tylko w czasie wykonywania programu. W obiektowych bazach danych natomiast mamy do czynienia z tym, iż obiekty zostają zapisane w sposób trwały w pamięci pomocniczej przed i po wykonaniu programu [15].

Obiektowe bazy danych nie stanowiły rozszerzenia oraz nie były ewolucją baz relacyjnych, stały się natomiast dla nich alternatywą. Relacyjne bazy danych w przeciwieństwie do obiektowych baz są uboższe semantycznie tj. do dyspozycji użytkownik dostaje ograniczoną ilość typów danych (tekst, data, liczby, współrzędne przestrzenne) oraz nie sprawdzają się w pewnych zastosowaniach np. w modelowaniu zagnieżdżeń wielopoziomowych. Ponadto model relacyjny zapewnia słabą reprezentację modelowanej dziedziny wynikającą z podziału encji lub klas na liczne relacje w procesie denormalizacji [15, 30]. Zasadniczą różnicą miedzy modelami jest to, iż w tradycyjnych bazach relacyjnych, dane oraz procedury operujące na tych danych są przechowywane oddzielnie (dane i relacje w bazie natomiast procedury w programie użytkownika) [11]. W przypadku modelu obiektowego dane w postaci składowych i procedury w postaci metod

są składowane i hermetyzowane [31] w jednym miejscu – obiekcie przechowywanym w bazie. Otrzymujemy więc w pełni semantyczny model danych, w którym z jednej strony mamy reprezentację danych za pomocą klas i składowych, natomiast z drugie strony otrzymujemy mechanizm do operowania na nich za pomocą metod.

Branże stosujące systemy z obiektowymi bazami danych to przede wszystkim [32, 33]: badania naukowe, transport, komunikacja, wydawnictwo, rozwój oprogramowania, lotnictwo, consulting, finanse, bankowość, zdrowie, medycyna, produkcja. Systemy bazodanowe do zarządzania danymi opartymi o model obiektowy określane są jako – Obiektowy System Zarządzania Bazą Danych OSZBD (*ang. object-oriented database management system - OODBMS*) lub (*ang. object database management system – ODBMS*).

Model strumieniowy

W modelu strumieniowym dane są przedstawione w postaci zbioru strumieni danych, w czasie rzeczywistym ciągłego, uporządkowanego, napływającego ciągu elementów (zdarzeń) w postaci danych do bazy [34]. W konwencjonalnym, relacyjnym modelu danych, są one modyfikowane w zbiorze danych jedynie na żądanie użytkownika. W świecie rzeczywistym istnieją jednak zjawiska, których próba opisu poprzez model relacyjny kończy się w praktyce niepowodzeniem. Związane jest to zazwyczaj z nieuwzględnieniem ciągłego i nieograniczonego w czasie i z czasem lawinowego napływu danych [34]. W związku z tymi i innymi ograniczeniami modelu relacyjnego [35] oraz w celu ich zniesienia powstała nowa koncepcja modelu i realizującego go systemu zarządzania danymi. Model ten jest oparty na aktywnej bazie danych (*ang. database active human passive – DAHP*). Do manipulacji danymi w DAHP wykorzystywany i implementowany jest język ciągłych zapytań oparty na SQL-u [36]. Prowadzone są różnego rodzaju prace nad przedstawieniem rozwiązania uniwersalnego w postaci kompleksowego Systemu Zarządzania Strumieniową Bazą Danych SZSBD (*ang. data stream management system – DSMS*), który służy do zarządzania danymi opartymi o model strumieniowy. Aktualnie rozwijanymi projektami z zakresu strumieniowych baz danych są [37-43]: Niagara, STREAM, Telegrach, OpenCQ, Hancock, MEDUSA, AURORA.

Model relacyjno-obiektowy

Model relacyjno-obiektowy (post relacyjny) umożliwia manipulowanie danymi jako zestawem obiektów, przy zachowaniu bazy relacyjnej z wewnętrznym mechanizmem przechowywania danych. Model relacyjno-obiektowy jest pierwszym modelem post relacyjnym, który rozszerza i uzupełnia braki m.in. ograniczoną ilość typów danych, występujących w modelu relacyjnym. Badaniami, które przyczyniły się do powstania tej

nowej grupy modeli i związanych z nimi systemów zarządzania bazami danych, stały się badania Michaela Stonebrakera [11]. W odróżnieniu od relacyjnego modelu danych nowe systemy posiadały dodatkowe możliwości w postaci [44]: obsługi rozszerzonych abstrakcyjnych typów danych (*ang. abstract data types – ATD*) definiowanych przez użytkowników, obsługi obiektów przez ustrukturyzowany język zapytań SQL i jego rozszerzenie w postaci obiektowego języka zapytań (*ang. object query language – OQL*) umożliwiające realizację mocnej kontroli typów czy dziedziczenia [45]. Wprowadzono też tzw. wyzwalacze czyli procedury uruchamiane automatycznie w przypadku zaistnienia zdefiniowanych przypadków. Mechanizm ten w dzisiejszych relacyjnych systemach baz danych jest powszechnie używany i stosowany.

Systemy do zarządzania danymi relacyjno-obiektowymi określane są jako Relacyjno-Obiektowych Systemów Zarządzania Bazą Danych ROSZBD (*ang. object relational database management system – ORDBMS*).

Model temporalny

Model temporalny stanowi odmianę modelu relacyjnego i strumieniowego z tego względu, że każdy rekord posiada stempel czasowy, określający czas, w jakim wartość jest prawdziwa. Tak więc oprócz danych „właściwych" dla danej modelowanej dziedziny przechowuje się też czas ważności tych danych z czasem transakcji. Czas ważności jest to czas kiedy zdarzenie (fakt) reprezentowane przez dane „właściwe" miało miejsce w opisywanej przez bazę rzeczywistości, zaś czas transakcji to czas kiedy dane są przechowywane w bazie danych (okres od ich wprowadzenia do ewentualnego usunięcia) [46, 47]. Językiem zapytań do bazy danych opartej o model temporalny jest język logiki temporalnej I rzędu [46] lub temporalny strukturalny język zapytań (*ang. temporal query language – TSQL2*) [48]. Oba podejścia bazują na algebrze relacyjnej i posiadają operatory algebry relacyjnej, które pozwalają operować na danych temporalnych (wyciągać historię).

System obsługi baz danych oparty na modelu temporalnym znajduje zastosowanie przy tworzeniu: aplikacji finansowych, aplikacji ubezpieczeniowych, systemów rezerwacji hoteli/biletów, systemów zarządzania danymi medycznymi, systemów wspierania decyzji. Przykładowymi temporalnymi bazami danych są m.in. TimeDB i Tiger [49, 50].

Model multimedialny

System zarządzania bazą danych oparty o model multimedialny umożliwia składowanie i analizę informacji w postaci obrazów, dźwięków, nagrań wideo i tekstu. Do podstawowych funkcji, które system ten powinien zawierać, należą m.in.: [51, 52]: rozwijanie formalnych technik modelowania semantycznego dla informacji multimedialnych

w szczególności dla danych wideo i obrazów, projektowanie elastycznych i mocnych metod indeksowania, wyszukiwania i organizowania danych multimedialnych, rozwijanie modelu przystosowanego do spełnienia wymagań takich, jak synchronizacja i integracja danych, implementacja i rozwijanie formalnego multimedialnego języka zapytań (*ang. multimedia query languages – MQL*), rozwijanie efektywnego składowania baz danych na fizycznych nośnikach danych, projektowanie i rozwijanie użytecznej architektury i wspierających ich systemów operacyjnych, zarządzanie rozproszonymi danymi multimedialnymi.

Systemy zarządzania multimedialnymi bazami danych SZMBD (*ang. multimedia database management system – MMDBMS*) budowane są najczęściej na bazie relacyjnych i relacyjno-obiektowych modeli lub ich systemów zarządzania, do których wprowadza się rozszerzenia multimedialne [51]. Powód stosowania relacyjnych, relacyjno-obiektowych systemów a nie specjalizowanych (dedykowanych) systemów tworzonych od podstaw nie jest przypadkowy. Liczba plików multimedialnych składowanych w takich systemach aktualnie sięga kilkudziesięciu milionów. Biorąc pod uwagę wyszukiwanie jedynie informacji alfanumerycznych w takich zbiorach, możliwości wydajnościowe przytaczanych SZBD są wystarczające [53].

Model natywny

Baza danych z natywnym modelem rozszerzalnego języka znaczników (*ang. extensible markup language – XML*) pozwala na przechowywanie danych wewnątrz bazy w postaci dokumentów XML [54]. Taki sposób zapisu ma duże znaczenie ze względu na popularność metajęzyka XML w wielu zastosowaniach, co wiąże się z jego bogatą semantyką, a więc i możliwością do elastycznego i wszechstronnego modelowania różnych zjawisk i zależności. Systemy obsługi baz danych oparte na modelu XML noszą nazwę – Native XML Database Systems. Nazywane są także semistrukturalnymi systemami zarządzania bazą danych. Informacje w tego rodzaju systemach są przetwarzane za pomocą zapytań wyrażonych w języku ścieżek rozszerzalnego języka znaczników (*ang. XML path language – XPath*) lub w języku zapytań rozszerzalnego języka znaczników (*ang. XML query language – Xquery*) [55]. Prezentacja treści i jej ewentualne transformacje odbywają się za pomocą przekształcenia rozszerzalnego języka arkuszy stylów (*ang. extensible stylesheet language transformations – XSLT*).

Model mobilny

Model mobilny składaja się z przenośnych baz danych, które są fizycznie odseparowane od serwera scentralizowanej bazy danych. Bazy zaprojektowane w oparciu o ten model mają możliwość komunikacji z serwerem z odległych lokalizacji. Dzięki temu pozwalają one na

współdzielenie danych organizacji. Rozwój tych baz jest następstwem rozwoju komunikacji bezprzewodowej [28].

Model hurtowni danych

Hurtownie danych (*ang. data warehause*) nazywane także magazynami danych, są tematycznie zorientowanymi podmiotowo, zintegrowanymi, zróżnicowanymi czasowo trwałymi kolekcjami danych przeznaczonych do wspomagania procesu podejmowania decyzji [28]. W hurtowniach danych, w przeciwieństwie do np. relacyjnych baz danych, wykorzystywana jest warstwa przetwarzania analitycznego (*ang. on-line analytical processing – OLAP*) a nie transakcyjnego (*ang. on-line transaction processing – OLTP*). (podpunkt 2.4.1 i 2.4.2). Aktualnie wysuwane są postulaty projektowania i tworzenia tzw. aktywnych hurtowni danych (*ang. active data warehouse*) łączących i wykorzystujących obie warstwy przetwarzania analitycznego (OLAP) i transakcyjnego (OLTP) [56]. Postulaty te wysuwane są ze względu na to, że procesy decyzyjne wymagają zazwyczaj oczyszczonych i odpowiednio przetransformowanych danych np. o różnego rodzaju trendach, które z natury zastosowań relacyjnego operacyjnego modelu mogą nie być tam przechowywane.

Hurtownie danych można podzielić na dwa rodzaje. Podział jest uwarunkowany wykorzystaniem przez nie modelu danych. Pierwszą grupę stanowią magazyny danych wykorzystujące model relacyjny, w których warstwę przetwarzania danych określa się jako relacyjny OLAP (*ang. relational OLAP – ROLAP*). Hurtownie danych typu relacyjnego budowane są według dwóch schematów: gwiazdy (*ang. star schema*) i płatka śniegu (*ang. snowflake schema*). Drugą grupę stanowią magazyny danych wykorzystujące wielowymiarowy model danych, w których warstwę przetwarzania określa się jako wielowymiarowy OLAP (*ang. multidimensional OLAP – MOLAP*). Hurtownie danych wykorzystujące wielowymiarowy model budowane są według schematu wielowymiarowej tablicy (*ang. multidimensional array*) [17].

Drugim wyznacznikiem podziału w przypadku hurtowni danych może być ich architektura przestrzenna (lokalizacja i liczba hurtowni). Mogą one występować w architekturze scentralizowanej lub sfederowanej tj. dane są zorganizowane logicznie, lecz fizycznie są przechowywane w osobnych bazach danych (magazynach danych) [17, 57].

2.2.3. Model silnie semantyczny

Model silnie semantyczny określany jest także jako model wiedzy zarządzany przez systemy zarządzania bazą wiedzy. Model semantyczny umożliwia modelowanie, przetwarzanie i operowanie na modelach zawierających głęboką, bogatą semantykę. Systemy

te oferują notację o dużej ekspresji służącą do reprezentowania faktów, więzów integralności, zapytań i funkcji modyfikacji [15].

Do grupy modeli silnie semantycznych należą: dedukcyjny model danych (*ang. deductive database system – DDS*) oraz model zorientowany koncepcyjnie nazywanym także modelem zorientowany na pojęcia (*ang. concept oriented model – COM*).

Dedukcyjny model danych

Dedukcyjny model danych (*ang. deductive database system – DDS*) związany jest z systemami ekspertowymi lub regułowo-modelowymi systemami ekspertowymi [2, 58]. Model ten obejmuje zastosowanie logiki formalnej do definiowania oraz operowania na danych i utrzymania ich integralności. Logika odgrywa tutaj rolę formalnych podstaw wyrażania, w jednolity sposób, pewnej liczby tych trzech aspektów technologii baz danych. Rozszerza ona również pojęcia konwencjonalnej bazy danych o narzędzia dedukcyjne [15]. Pierwsze badania nad zastosowaniem i związkiem logiki formalnej z bazami danych sięgają późnych lat siedemdziesiątych. Obecnie w modelach dedukcyjnych baz danych jako język „zapytań" operacyjny używa się Datalogu będącego pochodną języka Prolog używanego w zagadnieniach sztucznej inteligencji (*ang. artificial intelligence – AI*) [59].

Model zorientowany koncepcyjnie

Model zorientowany koncepcyjnie lub model zorientowany na pojęcia (*ang. concept oriented model – COM*) zaproponowany został przez Alexandra Savinova w 2004 [60]. Model ten stanowi nowe podejście do modelowania danych i bazuje na trzech głównych zasadach [61, 62]: zasadzie dwoistości (*ang. duality principle*) mówiącej, że każdemu elementowi (pojęciu) przypisana jest tożsamość (*ang. identity*) oraz encja (*ang. entity*), zasadzie włączenia (*ang. inclusion principle*) dotyczącej używania hierarchicznej struktury dla modelowania tożsamości oraz zasadzie porządku (*ang. order principle*), która mówi o używaniu matematycznej zasady porządku częściowego (*ang. partial order*) do reprezentowania semantyki danych.

Głównym celem nowo powstałego modelu opartego o ww. zasady jest dostarczenie prostego i efektywnego sposobu do reprezentacji i manipulacji wielowymiarowymi i hierarchicznymi danymi z zachowaniem możliwości do reprezentowania danych fizycznie [63]. Odróżniającym go elementami od innych modeli jest to, iż bazuje na teorii porządku zbiorów (*ang. order theory set*), w szczególności na matematycznej zasadzie porządku. W dużej mierze COM został zainspirowany przez teorie sieci (*ang. lattice theory*) [64] i formalną analizę pojęć (*ang. formal concept analysis – FCA*) [65] wprowadzoną (po raz pierwszy użyte pojęcie) przez Rudolfa Willea w 1984 roku a zbudowaną na teorii sieci

i częściowego porządku, które zostały rozwinięte przez Birkhoff i innych w latach 30 XIX wieku. Podobieństwo między COM a FCA tkwi w tym iż używają silnego matematycznego formalizmu w postaci sieci (*ang. lattice*), które są bogate w interpretacje semantyczne. Różnica polega na tym iż FCA zostało zaprojektowane po to aby używać go w obszarach takich, jak: nabywanie wiedzy, klasyfikacja i przetwarzanie informacji. Natomiast COM przeznaczony jest bardziej do efektywnego modelowania i przechowywania danych [61].

2.3. Liczba węzłów Systemów Zarządzania Bazą Danych

Przez liczbę węzłów należy rozumieć liczbę jednostek w systemie informatycznym, na których możliwe jest przechowywanie i przetwarzanie danych, informacji czy też wiedzy. Ze względu na liczbę węzłów mówi się o bazach scentralizowanych (podrozdział 2.3.1) oraz rozproszonych (podrozdział 2.3.2).

2.3.1. Bazy scentralizowane

Bazy scentralizowane zawierają jedną centralną bazę danych z systemem jej zarządzania, do której przychodzą zgłoszenia. Zgłoszenia stanowią żądania pochodzące z aplikacji klienckiej w architekturze dwuwarstwowej bądź z aplikacji serwerowych w architekturze wielowarstwowej [66-68].

2.3.2. Bazy rozproszone

Baza rozproszona lub architektura rozproszona zarządzania bazami danych występuje wówczas, gdy baza danych istnieje fizycznie na dwóch lub większej liczbie komputerów i traktowana jest jak jedna logiczna całość. Dzięki takiemu podejściu, zmiany w zawartości bazy w jednym komputerze są uwzględniane również w innych maszynach. Z punktu widzenia użytkownika wszystkie te bazy logicznie stanowią jedną bazę danych. Rozproszone bazy danych są stosowane ze względu na zwiększoną wydajność przetwarzania na wielu komputerach jednocześnie oraz na to iż znoszą pewne ograniczenia baz scentralizowanych [7, 12, 69].

2.4. Warstwa przetwarzania w Systemach Zarządzania Bazą Danych

Przez termin *warstwa przetwarzania* należy rozumieć technologie, techniki i algorytmy, za pomocą których dokonuje się operacji na danych zebranych w bazie danych poprzez SZBD lub poprzez oddzielne, wyspecjalizowane do tego zewnętrzne programy. Działania te mają na celu dostarczenie użytkownikowi informacji, wiedzy na interesujący go

temat. Do warstw przetwarzania w SZBD należą: warstwa aplikacji ciągłego przetwarzania transakcyjnego (podrozdział 2.4.1), warstwa aplikacji przetwarzania analitycznego (podrozdział 2.4.2) oraz warstwa eksploracji danych (podrozdział 2.4.3).

2.4.1. Warstwa aplikacji ciągłego przetwarzania transakcyjnego

Narzędzia do transakcyjnego przetwarzania ciągłego wbudowane są najczęściej w SZBD i mają za zadanie na bieżąco obsługiwać transakcje dokonywane przez użytkowników w systemie. SZBD oparty o transakcje posiada mechanizm ich obsługi umożliwiający obsługiwanie wielu użytkowników w jednym czasie bez względu na ich lokalizację czasoprzestrzenną.

Transakcje obsługiwane przez OLTP powinny spełniać wymagania atomowości, spójności, izolacji i trwałości (*ang. atomicity, consistency, isolation, durability – ACID*) [70]. Atrybutem OLTP opatrywane są operacyjne bazy danych. Systemy OLTP ukierunkowane są na realizację operacji: aktualizacji, tworzenia, zastępowania i kasowania (*ang. create, replace, update, delete – CRUD*) danych [70].

2.4.2. Warstwa aplikacji przetwarzania analitycznego

Narzędzia do ciągłego przetwarzania analitycznego używają wielowymiarowych perspektyw zagregowanych danych w celu zapewnienia szybkiego dostępu i obsługi strategicznych informacji przeznaczonych do zaawansowanych analiz. Technologia OLAP pozwala użytkownikom na uzyskanie głębszego zrozumienia oraz dodatkowej wiedzy o różnorodnych aspektach danych organizacji. Możliwe jest to poprzez szybkie, konsekwentne i interakcyjne dostępy do wielu odmian możliwych sposobów widzenia danych [71].

2.4.3. Warstwa eksploracji danych

Eksploracja danych (*ang. data mining – DM*) stanowi interdyscyplinarną dziedzinę naukową, która wywodzi się z dziedziny nauki i techniki jaką jest informatyka. Zajmuje się ona wizualizacją i wydobywaniem ukrytej *implicite* informacji z dużych zasobów informacyjnych (baz danych) [72]. Wykorzystuje w tym celu technologie przetwarzania informacji oparte o statystykę i sztuczną inteligencję: uczenie maszynowe, metody ewolucyjne, logikę rozmytą oraz zbiory przybliżone. Nieodłącznie związana jest z różnego rodzaju omawianymi w tym artykule źródłami i modelami danych. Ogólnie eksploracja danych (ED) przeprowadzana jest na bardzo dużych bazach danych w celu wyekstrahowania z nich wartościowej wiedzy. Eksploracja danych ukierunkowana jest na zastosowania.

Problemy badawcze a zatem też zastosowania metod i algorytmów z zakresu ED, które za jej pomocą się podejmuje, są mocno zależne od specyfikacji dostępnych zbiorów danych dotyczących świata rzeczywistego [20, 73]. Na potrzeby eksploracyjne budowane są także specjalne platformy i języki pośredniczące, wspomagające eksplorację danych. Do takich rozwiązań pośredniczących (*ang. middleware*) zalicza się język znaczników odkrywania wiedzy w bazach danych (*ang. knowladge discovery in databse markup language – KDDML*). Platforma ta i wspierany przez nią język została zaprojektowana w celu wspierania rozwoju finalnych aplikacji lub wysoko poziomowych systemów, które wykorzystują: mieszany dostęp do danych, wstępne przetwarzanie danych, ekstrakcję oraz wdrażanie modeli eksploracji danych. Cała platforma została oparta o strukturę języka znaczników XML [74, 75].

2.5. Aktywność Systemów Zarządzania Bazą Danych

Poprzez termin *aktywność* należy rozumieć zdolność lub brak zdolności do dokonywania dodatkowych operacji w obrębie bazy danych lub systemu zarządzania. Do przeprowadzania dodatkowych operacji służy mechanizm przetwarzania danych wykonywany poza główną sesją żądań użytkownika. Umożliwia on np. wykonywanie dodatkowych operacji na danych poza kwerendą (zapytaniem) wysłaną do SZBD. Ze względu na aktywność bazy danych możemy podzielić na aktywne (podrozdział 2.5.1) oraz nieaktywne (podrozdział 2.5.2).

2.5.1. Systemy aktywne

Termin *aktywny system* określa grupę platform zarządzania bazą danych, które pozostają czynne nawet wtedy, gdy nie są do nich jawnie kierowane żądania transakcyjne czy zadania tworzone przez użytkownika. Możliwa jest więc zmiana stanu poza główną sesją użytkownika i najczęściej występuje ona na skutek [76]: zajścia określonego zdarzenia zewnętrznego lub wewnętrznego, zakończenia realizacji określonego zbioru transakcji kierowanych do SZBD, upływu określonego kwantu czasu lub kombinacji dwóch powyższych przypadków.

W odróżnieniu od opisanych w następnym podpunkcie systemów nieaktywnych (pasywnych), systemy aktywne baz danych są wyposażone w mechanizm definiowania i przetwarzania aktywnych reguł (*ang. event condition action – ECA*), których logika inspirowana była regułami z systemów eksperckich [14, 77]. Wykorzystuje się zatem operatory zdarzeniowe, modele aktywności i zależności czasowych oraz przyczynowo skutkowe do obsługi zdarzeń powstających między akcjami użytkownika.

2.5.2. Systemy nieaktywne

Nieaktywne bazy danych nazywane także pasywnymi bazami lub pasywnymi systemami baz danych charakteryzują się tym, iż nie wykonują żadnych dodatkowych zadań np. związanych z zapytaniem użytkownika do bazy danych. Nie posiadają one także jakiegokolwiek mechanizmu mogącego takie dodatkowe zadania obsługiwać.

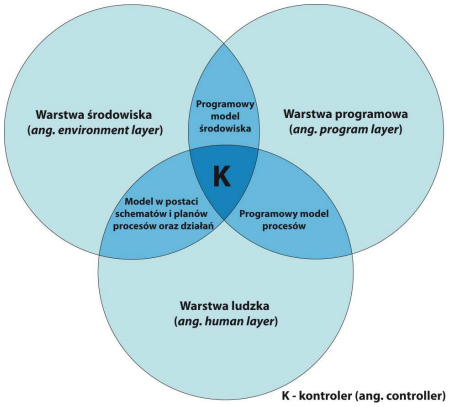
3. Zarys projektu platformy agentowej dla Państwowej Straży Pożarnej

W służbach ratowniczych PSP istnieje problem w stworzeniu jednolitego, funkcjonalnego, komputerowego systemu wspomagania decyzji z elementami edukacyjno-szkoleniowymi, przeznaczonego dla Kierującego Działaniami Ratowniczymi (KDR) [78]. Dotychczas w szkole głównej PSP opracowano kilka przykładowych rozwiązań i koncepcji. Rozwiązania te dotyczyły głównie: tworzenia modelu i architektury systemu [79], opisu jego wymagań i specyfikacji [78] oraz wyboru metod opartych o wyznaczenie, analizę i rozwiązania problemów występujących na początkowych etapach realizacji systemu [80]. Ze względu na występujące problemy, wynikające m.in. ze złożoności analizowanej dziedziny, dotychczas nie powstał jeszcze działający, w pełni funkcjonalny prototyp, w dalszym ciągu są to badania laboratoryjne. Dodatkowo pojawiają się kwestie dotyczące tego, jak zorganizować architekturę sprzętowo-programową systemu, jakie technologie wybrać do tego celu, w jaki sposób reprezentować i pozyskiwać wiedzę w systemie oraz jaki proces budowy systemu wybrać. Problematyka procesu budowy obejmuje wymienione kwestie jak i problem dotyczący tego czy znajdą się odbiorcy na tego rodzaju rozwiązanie i wyniki analiz.

W niniejszym rozdziale skupiono się na opisie i rozwiązaniu pierwszego wymienionego problemu, który stanowi dobór architektury sprzętowo-programowej i technologicznej. W szczególności skoncentrowano się w tej sekcji artykułu na opisie zastosowania koncepcji architektury agentowej (podpunkt 2.1.3). Dokonano analizy możliwości jej zastosowania w służbach ratowniczych PSP. Zaproponowano także podstawową strukturę aplikacyjną do rozwiązywania problemów związanych z ciągłym napływem danych. Dane te napływają z akcji ratowniczo-gaśniczej od jej zgłoszenia do dyspozytora przez obserwatora do jej rozwiązania przez KDR – model transakcyjny. W następnej kolejności zaproponowano zastosowanie architektury agentowej do rozwiązywania zadań analitycznych, do których należą np. analiza zużycia sprzętu, jego zakup etc. A więc zadań długofalowych - model analityczny.

3.1. Transakcyjna architektura agentowa

Przed omówieniem konkretnej realizacji architektury agentowej dla służb ratowniczych PSP, opisano relacje jakie zachodzą pomiędzy trzema podstawowymi warstwami z jakimi projektant oprogramowania styka się podczas jego projektowania. Warstwami tymi są: środowisko (*ang. environment layer*), oprogramowanie (*ang. program layer*) oraz ludzie (*ang. human layer*). Ich wzajemne relacje wyznaczają obszary modelowanej na pewnym wybranym poziomie koncepcji, wybranej części rzeczywistości. Poszczególne warstwy wraz z możliwymi relacjami przedstawia Rycina 2.



Ryc. 2 Podział rzeczywistości projektanta oprogramowania na warstwy ze względu na możliwości jej modelowania

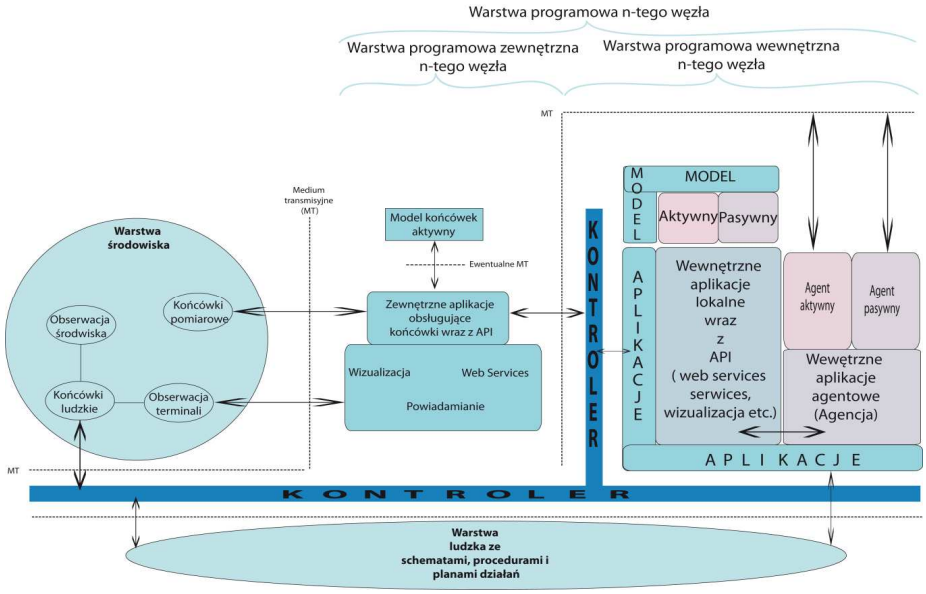
Źródło: Opracowanie własne

Rycina 2 przedstawia podział rzeczywistości projektanta oprogramowania na warstwy ze względu na możliwości jej modelowania. Na wymienionej rycinie, okręgi wyznaczają warstwy, które biorą udział w modelowaniu, natomiast przecięcia się okręgów wyznaczają powstałe modele w wyniku interakcji warstw. Część wspólną stanowi *kontroler*, którego zadaniem jest nadzorowanie przepływu danych, informacji, wiedzy między poszczególnymi warstwami oraz modelami i reagowanie na zachodzące zmiany. Na przedstawionym rysunku *warstwę środowiska* tworzą wszelkie możliwe interakcje i procesy, jakie zachodzą w świecie rzeczywistym pomiędzy wszystkimi jego uczestnikami. Programowe modelowanie wybranego elementu środowiska np. pomiaru poziomu wody, zapisu go w bazie danych i dalszej obróbce programowej, tworzy tzw. *programowy model środowiska*. Jeśli dla wybranego elementu środowiska, dotyczącego obszaru działań człowieka, tworzone są formalne procedury czy też schematy działań wyrażone np. za pomocą planów ratowniczych, wówczas tworzone są tzw. modele w postaci heurystyk, schematów i planów procesów oraz

działań. Wdrożenia ich dokonuje się w warstwie ludzkiej w celu znormalizowania procesów związanych z komunikacją czy przeprowadzaniem akcji ratowniczo-gaśniczej.

Warstwa ludzka jest wyodrębnionym, pod pewnym względem, znormalizowanym i ujednoliconym wycinkiem rzeczywistości, w której zachodzą interakcje między ludzkie ograniczone przez wyznaczone normy i procedury. Jeśli istnieje możliwość komputerowego, programowego wsparcia utworzonych procesów w warstwie ludzkiej wówczas otrzymywany jest tzw. programowy model procesów, który może zostać wykorzystany do wspomagania ludzi, w tym kontekście nazywanych KD, w ich decyzjach. W takim przypadku systemy wspierające lub ułatwiające i automatyzujące pewne aspekty procesu zachodzącego w warstwie ludzkiej nazywane są powszechnie systemami wspomagania decyzji lub systemami wspomagania pracy.

Wyżej opisany model warstwowy można zaadoptować na potrzeby np. modelowania systemu wspomagania decyzji z rozproszoną warstwą programową dla służb ratowniczych PSP. Schemat implementacji z wyróżnieniem stosu programowego w warstwie programowej przedstawia Rycina 3.



Ryc. 3 Dynamiczne oddziaływanie pomiędzy warstwami z wyszczególnieniem stosu programowego w warstwie programowej *n-tego* węzła

Źródło: Opracowanie własne

Rycina 3 przedstawia dynamiczne oddziaływanie pomiędzy warstwami z wyszczególnieniem stosu programowego w warstwie programowej *n-tego* węzła. Przez

termin *węzeł* należy rozumieć fizyczną lokalizację sprzętowo-programową w systemie przestrzennym np. w komendzie Białostockiej, Warszawskiej etc. W warstwie środowiska zostały umieszczone tzw. *końcówki*. Końcówki stanowią sensory: ludzkie oraz pomiarowe (analogowe, cyfrowe, analogowo-cyfrowe). Sensory te nadzorują wybrane parametry otoczenia, w którym umieszczone są *końcówki pomiarowe* np. wskaźników poziomu wody. *Końcówkami ludzkimi* mogą być np. osoby obserwujące bezpośrednio poziom wody na rzece lub odczytujące jej poziom z terminali programowych, wizualizujących jej poziom lub dostarczających informacji o jej konkretnej wartości w wybranym punkcie. Rejestracja danych za pomocą *końcówek pomiarowych* najczęściej odbywa się przez system akwizycji danych pomiarowych, który wchodzi w skład *warstwy programowej* i dodatkowo w omawianym kontekście stanowi tzw. *warstwę programową zewnętrzną*. Zbudowany jest on najczęściej z bazy danych, w której rejestrowane dane pomiarowe, zapisywane i zarządzane są przez wybrany system zarządzania bazą danych, (model końcówek aktywny). Dane z końcówek są odczytywane i zapisywane przez zewnętrzne aplikacje. Aplikacje te zazwyczaj dostarczane są także z ich interfejsem programowania aplikacji. Dodatkowymi dostarczanymi rozwiązaniami w omawianej warstwie mogą być:

- system wizualizacji, który służy do graficznej reprezentacji rejestrowanych zmiennych,
- system powiadamiania, który służy do zwracania uwagi dyspozytorowi na wybrane stany systemu, dzięki czemu może on podjąć odpowiednie działania,
- usługi sieciowe (*ang. web service*), dzięki którym możliwa jest zdalna komunikacja z systemem przez niezależne aplikacje pisane przez niezależnych programistów.

Termin *programowa warstwa zewnętrzna* oznacza, że aplikacje te nie wchodzą bezpośrednio w skład warstwy oprogramowania dla służb ratowniczych PSP. Są one dostarczane przez zewnętrznych dostawców oprogramowania i działają niezależnie od systemu wspomagania decyzji. Dzięki swojemu API i usługom sieciowym mogą działać jak wtyczki (*ang. plugins*) i tym samym rozszerzać funkcjonalność wspomnianej platformy. W przypadku, gdy któraś z obserwowanych przez końcówki ludzkie lub pomiarowe wartości zmiennych środowiska zostanie przekroczona, wówczas uruchamiane są przewidziane procedury w warstwie ludzkiej. Przykładową sytuację może stanowić scenariusz przypadku (*ang. event case*), w którym obserwator na rzece, lub dyspozytor na stacji pomiarowej poziomu wody, dostrzega znaczne przekroczenie poziomu lub o takim stanie zostaje

poinformowany przez system monitujący. Stan taki grozi zalaniem terenów, w takiej sytuacji dyspozytor uruchamia procedurę działania przeciwko powodzi i zawiadamia odpowiednie jednostki służb ratowniczych. Wybrane jednostki mogą zostać już wcześniej powiadomione (system wczesnego powiadamiania) o zaistniałym zagrożeniu, dzięki czemu mogą odpowiednio przyszykować się do działania np. opracowując odpowiednią strategię działań ratowniczych. Wcześniejsze powiadomienie odbywa się na drodze komunikacji *warstwy programowej zewnętrznej* z *warstwą programową wewnętrzną* stanowiącą właściwe oprogramowanie dla służb ratowniczych PSP. Dokładniej, komunikacja ta zachodzi na poziomie usług sieciowych *warstwy programowej wewnętrznej*. Wówczas programiści odpowiedzialni za zewnętrzne aplikacje dopisują rozszerzenia komunikujące się z systemem PSP, w których przesyłają komunikaty o stanie nadzorowanego obiektu. Możliwa jest także sytuacja odwrotna, w której to programiści wewnętrzni dopisują rozszerzenia pomiarowe, monitorujące stan wybranych obiektów, wykorzystując ich zewnętrzne usługi sieciowe z ich API. Po dotarciu zawiadomienia, żądania (*ang. request*) do kontrolera zarówno programowego jak i ludzkiego, w postaci np. dyspozytora, i wstępnym rozpoznaniu sytuacji za pośrednictwem wszelkich możliwych dostępnych sensorów, wyzwalany jest proces akcji ratowniczo-gaśniczej. Etapy procesu przeprowadzania akcji ratowniczo-gaśniczej przez KDR mogą być wspierane przez *warstwę programową wewnętrzną*. W jej skład wchodzą aplikacje komunikujące się z modelem zawierającym tzw. *część aktywną* oraz *pasywną*. W *części aktywnej* zawarte są wszelkie informacje dotyczące bieżącego stanu systemu np. ilości wolnych i zalokowanych zasobów (sprzętu i ludzi), ich miejsca zalokowania jak i najszybszych tras dojazdu do miejsca zdarzenia. Z części tej mogą także pochodzić zdjęcia satelitarne obiektu którego dotyczą działania ratowniczo-gaśnicze lub przestrzeni, na której rozgrywa się dane zdarzenie. Dodatkowo model ten może być uzupełniany informacjami z modeli zewnętrznych np. z dostępnych czujników itd. W *modelu pasywnym* znajdują się archiwalne raporty z odbytych zdarzeń ratowniczo-gaśniczych. Zorganizowane i opisane są one według wybranej ontologii wyrażonej np. w technice obiektowej [20, 31]. Wewnętrzny system aplikacji odpowiedzialny jest za wypracowanie jak najlepszej odpowiedzi dla KDR, w postaci wskazówek. Odpowiedź uzyskuje się na podstawie dopasowania informacji z *modelu aktywnego* oraz informacji dostarczonych z miejsca zdarzenia przez KDR, do modelu pasywnego. Dopasowanie polega na wyszukaniu informacji (*ang. information retrieval – IR*) historycznej. Informacja ta ma postać najbardziej podobnego przypadku zdarzenia, który zawarty jest w *modelu pasywnym*.

Przełączenie na tryb agentowy następuje w przypadku, gdy:

- powstałe zagrożenie jest na tyle duże, iż swym zasięgiem obejmuje obszar lub obiekt w taki sposób, że zadysponowane siły i środki są niewystarczające do jego neutralizacji,
- nie można wyszukać odpowiedniego, dającego najlepsze wskazówki raportu archiwalnego w węźle lokalnym aplikacji na podstawie dostarczanych do niego aktualnych danych.

Tryb ten polega na tym, iż w wydzielonej części systemu informatycznego tworzy się aplikację programową tzw. agencję której rolą jest:

- nadzór nad agentami m.in. powoływanie ich do życia oraz przyjmowanie ich z innych systemów,
- przekazywanie wyników działań agentów do wewnętrznej aplikacji lokalnej, która wysyła je do dowódców akcji.

Agenci są więc niezależnymi programami mogącymi przemieszczać się między wybranymi węzłami systemu, według wybranej strategii. Agencja podzielona jest na dwie części. Pierwsza zawiera *agentów aktywnych*. Druga natomiast *agentów pasywnych*. Agenci aktywni i pasywni komunikują się poprzez agencję węzła, do którego zostali wysłani z jego częścią aktywną natomiast agenci pasywni z jego częścią pasywną. W ten sposób możliwe jest pozyskanie informacji na temat m.in.: dodatkowych sił i środków, które mogą zostać zadysponowane na miejscu zdarzenia w razie zajścia takiej konieczności oraz sposobów likwidowania zdarzenie, poprzez przeszukiwanie modelu pasywnego wybranego węzła.

3.1.2. Realizacja transakcyjnej architektury agentowej

Przykładowa realizacja transakcyjnej architektury agentowej opisanej w punkcie 3.1 została przedstawiona na Rycinie 4.

Ryc. 4 Przykładowa realizacja architektury agentowej dla warstwy programowej składającej się z trzech węzłów głównych

Źródło: Opracowanie własne

Rycina 4 przedstawia realizację architektury agentowej dla warstwy programowej składającej się z trzech węzłów głównych. Węzły te zawierają opisywaną w punkcie 3.1. konfigurację programową składającą się z: aplikacji (serwer aplikacji), modelu danych (serwer baz danych) oraz agencji (serwer agentowy). Do przykładowych węzłów na stopniu wojewódzkim należą: komenda wojewódzka w Warszawie, Białymstoku oraz Olsztynie. W przykładzie, do komend wojewódzkich kierowane są żądania obsługi akcji ratowniczo-gaśniczych z obszaru całego województwa oraz z obszarów podległych komendom miejskim i powiatowym. Do systemu mają więc dostęp Miejskie Stanowiska Kierowania MSK jak i Powiatowe Stanowiska Kierowania PSK. Wszystkie komendy wojewódzkie rejestrują działalność służb ratowniczych na wybranym terenie (model aktywny) jak i pozyskują opisy zneutralizowanych zdarzeń, w których brały udział siły i środki podległe komendom miejskim i powiatowym (model pasywny). Komendy niższego szczebla, miejskie i powiatowe, nie przechowują fizycznie danych i całej warstwy serwerowej. Łączą się one natomiast z warstwą aplikacji poprzez terminale w celu pozyskania lub dostarczenia odpowiedniej informacji, wiedzy o zaistniałym zdarzeniu i przeprowadzonej akcji ratowniczo-gaśniczej. W przypadku gdy w węźle na stopniu wojewódzkim nie uda się odnaleźć satysfakcjonującego rozwiązania, wówczas przeszukiwane są najbliższe węzły pod

względem odległości jak i profilu zdarzeń. Zdarzenia np. w województwie podlaskim stanowią głównie pożary torfowisk i lasów [81], z tego względu powinny być przeszukane najbliższe komendy wojewódzkie, o zbliżonym profilu, które przechowują największą ilość zdarzeń tego typu interwencji [7]. Profile zdarzeń w postaci metadanych mogą być utrzymywane w Komendzie Głównej (KG). Trasa agenta (*ang. routing*), w takim przypadku, może wyglądać następująco: agent udaje się do warstwy aplikacji KG, z której kierowany jest do warstwy aplikacji najbliższej komendy wojewódzkiej po czym po pobraniu danych wraca bezpośrednio do węzła aplikacji, z którego został wysłany.

3.2. Analityczna architektura agentowa

Analityczna architektura agentowa konfiguracją sprzętowo-programową nie różni się od architektury transakcyjnej opisanej w podpunkcie 3.1. Różnice są funkcjonalne, architektura transakcyjna przeznaczona jest do obsługi akcji w trybie *on-line* i aktywnego podsuwania rozwiązań dla KDR. Architektura analityczna przeznaczona jest natomiast do badania zaistniałych zdarzeń np. pod kątem zużytych sił i środków w czasie akcji czy też planowania zakupu nowego sprzętu. Zakup nowego sprzętu specjalistycznego może być podyktowany analizą:

- zgromadzonych informacji o zużyciu sprzętu i zapotrzebowania na nowy (model pasywny),
- informacji o sprzęcie na rynku oferowanych przez zewnętrznych dostawców (model aktywny) według profilu i częstotliwości występowania zdarzeń w danym rejonie.

Zewnętrzny model aktywny, który należy do *warstwy programowej zewnętrznej n*-tego węzła, może być reprezentowany przez elektroniczną ofertę firmy specjalizującej się w dostarczaniu specjalistycznego sprzętu ratowniczo-gaśniczego. Oferty mogą być dostępne poprzez serwisy internetowe. Analizą ich zajmują się wówczas aktywni agenci typu analitycznego, którzy aktywnie poszukują jak najlepszej oferty, pod względem cenowym i wymogów. Aktualnie próbę takiego rodzaju systemów podejmuje się w obrębie systemów handlu elektronicznego (*ang. e-commerce*) np. do negocjacji cen [25]. Podobne rozwiązania mogą zostać zastosowane w obrębie omawianego modułu analitycznego do wyboru jak najlepszej oferty.

4. Podsumowanie

Proces projektowania i implementacji zintegrowanego systemu, na poziomie warstwy programowej, dla służb ratowniczych PSP jest zadaniem skomplikowanym. Świadczyć o tym może fakt złożoności samej analizowanej dziedziny. W obrębie działań PSP obsługuje się bowiem zdarzenia o różnym charakterze np. pożary, zagrożenia miejscowe czy też fałszywe alarmy. Każde z tych zdarzeń wymaga innego sposobu i planów reagowania oraz użycia sprzętu. Ponadto dla każdego rodzaju interwencji i zadań z nimi związanych powinny zostać opracowane struktury danych, w postaci baz danych (modeli pasywnych oraz aktywnych) będacych własnością PSP, z którymi będą mogły komunikować się, dzięki *wewnętrznej wartwie aplikacji*, dowolne aplikacje spełniające kryteria integralności i bezpieczeństwa. Komunikacja powinna odbywać się także w drugą stronę np. z czujnikami monitorującymi istotne zmienne dla newralgicznych obiektów oraz obszarów. Komunikacja w obu kierunkach może odbywać się poprzez spełnienie kontraktu programowego [31] i wykorzystanie technik oraz protokołów komunikacyjnych w postaci [82-84]: protokołu wywoływania zdalnego dostępu do obiektów (*ang. simple object access protocol – SOAP*), zdalnego wywołania procedur (*ang. remote procedure call – RPC*) czy architektury zapewniającej komunikację pomiędzy obiektami pracującymi w heterogenicznych systemach komputerowych (*ang. common object request broker architecture – CORBA*). W przypadku tworzenia jednolitego oprogramowania dla aplikacji *węzłów wewnętrznych* można oprzeć się na technologi Java oraz dostarczanej wraz z nią technik zdalnego wywoływania metod (*ang. remote method invocation – RMI*) [83] czy też szkieletu aplikacji do rozwijania oprogramowania agentowego (*ang. java agent dEvelopment framework – JADE*) [85]. Przy opracowaniu odpowiedniej konfiguracji, system może pracować też jako platforma do wczesnego ostrzegania np. przed powodziami. Konfiguracja taka wymagałaby utworzenia strategi komunikacji między węzłami ich połączenie i skonfigirowania. Otwartą kwestią pozostaje sposób rozmieszczenia węzłów oraz ich ilość. W przedstawianym opracowaniu założono, że Komendy Wojewódzkie przechowują cały rejestr, związany z opisem zdarzeń (model pasywny) i z nadzorem dysponowanych zasobów (model aktywny), pochodzący z komend podległych tj. miejskich oraz powitowych. Taka konfiguracja może prowadzić do przeciążeń systemu (*ang. overloading*). Przeciążenie następuje, gdy do systemu zaczynają napływać zgłoszenia, w ilości której nie będzie on w stanie przetworzyć. Jednym z możliwych wówczas rozwiązań jest replikacja oraz skalowanie sprzętowe. Dokonać można rozbudowy mocy obliczeniowej jednostek, do których są kierowane żądania i projektowanie

ich na najwyższe możliwe obciążenia, które z góry ciężko jest przewidzieć i oszacować. Drugim wyjściem może być skalowanie poprzez większe rozproszenie aplikacji, a więc instalacji na stopniach powiatu i gminy podobnego oprogramowania jak na stopniu wojewódzkim, przez co serwer wojewódzki zostaje odciążony. W przypadku tym zostaje skomplikowana struktóra aplikacyjna przez co problem kontroli i utrzymania całej aplikacji we wszystkich węzłach staje się bardziej kłopotliwy. Architektura taka zwiększa również koszty – należy zakupić dodatkowy sprzęt i oprogramowanie. Można postawić hipotezę, że struktóra rozproszona jest optymalna i efektywna ze względu na proces przeszukiwania i wyszukiwania przypadków zdarzeń oraz sprawuje się lepiej w przypadku dużego obciążenia systemu związanego np. z wysyłaniem do niego żądań, związanych z pojawieniem się dużej ilości porzarów lasów na terenie województwa. Obecnie w Zakładzie Informatyki i Łączności SGSP prowadzone są badania nad zastosowaniem architektury rozproszonej baz danych w służbach ratowniczych PSP [6, 12, 86, 87].

Aktualnie w warstwie ludzkiej istnieje rozbudowana baza heurystyczna, na którą składają się dobrze udokomunentowane precesy, regulaminy, plany i akty prawne [88-90]. Wszytkie one wyznaczają zakres kompetecji i działań odpowiednich jednostek organizacyjcnych PSP takich jak np. Komend Wojewódzkie, miejskie, powiatowe czy też stanowisk kierowania na różnym szczeblu. Stanowią one zatem dobry punkt wyjściowy do projektowania, budowy i organizacji nowoczesnej oraz sprawnej, opisywanej w niniejszym artykule warstwy aplikacyjnej w postaci systemu do obsługi zdarzeń i związanych z nimi akcji ratownoczo-gaśniczych dla służb ratowniczych PSP.

Literatura

1. Nonaka I., Takeuchi H., *Kreowanie wiedzy w organizacji,* Poltext, Warszawa, 2002;
2. Sostaric B., *Przeglądowa analiza systemów do zarządzania wiedzą z elementami systemów ekspertowych,* Studia i materiały polskiego stowarzyszenia zarzadzania wiedza, Bydgoszcz 2005. s. 150 - 158;
3. Kuc B. R., Żemigała M., *Zarządzanie wiedzą - postulat czy fakt? Inżynieria wiedzy i systemy ekspertowe,* Akademicka Oficyna Wydawnicza EXIT, Warszawa 2009;
4. Krasuski A., Kreński K., *Ewid 9x i co dalej ?* Przegląd Pożarniczy, nr 6, 2006;
5. Strona firmy abakus. [on-line] [dostęp: 1 marca 2009] Dostępny w Internecie: http://www.ewid.pl/?set=main&gr=aba;
6. Krasuski A., Maciak T., *Rozproszone bazy danych, możliwości ich wykorzystania w Państwowej Straży Pożarnej,* Zeszyty Naukowe SGSP, nr. 34, 2006, s. 23-42;

7. Krasuski A., Maciak T., *Wykorzystanie rozproszonej bazy danych oraz wnioskowania na podstawie przypadków w procesach decyzyjnych Państwowej Straży Pożarnej,* Zeszyty Naukowe SGSP, nr. 36, 2008, s. 17-35;
8. Mirończuk M., Kreński K., *Koncepcja systemu ekspertowego do wspomagania decyzji w Panstwowej Strazy Pozarnej*. [w:] Grzech A., Juszczyn K., Kwaśnicka H. and Nguyen N.T., editors. Inżynieria Wiedzy i Systemy Ekspertowe, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2009;
9. Kempa A., *Modelowanie procesów biznesowych z wykorzystaniem metody Case-Based Reasoning,* Studia i materiały polskiego stowarzyszenia zarzadzania wiedzą. Bydgoszcz 2005, s. 50-56;
10. Maciak T., Kreński K., *Dobór atrybutów bazy przeciwpożarowej budynków systemu informacji przestrzennej służb ratowniczych,* Zeszyty Naukowe SGSP, nr 32, 2005, s. 59;
11. Krasuski A., Maciak T., *Historia rozwoju Systemów zarządzania bazami danych,* Bezpieczeństwo i Technika Pożarnicza, Wydawnictwo CNBOP Józefów 2006, s. 213-226;
12. Krasuski A., Maciak T., *Rozproszone bazy danych - architektura funkcjonalna,* Bezpieczeństwo i Technika Pożarnicza, nr. 01/ 2007, Wydawnictwo CNBOP, Józefów 2007;
13. Kreński K., Maciak T., Krasuski A., *An overview of markup languages and appropriateness of XML for description of fire and rescue analyses,* Zeszyty Naukowe SGSP, nr. 37, 2008, s. 27-39;
14. Królikowski Z., *Wprowadzenia do kursu: Zawansowane systemy baz danych,* [on-line] [dostęp: 27 sierpnia 2009] Dostępny w Internecie: http://osilek.mimuw.edu.pl/index.php?title=Zaawansowane_systemy_baz_danych.
15. Beynon-Davies P., *Systemy baz danych,* Wydawnictwa Naukowo-Techniczne, Warszawa 2003;
16. Morawski O., *Hurtownie danych i systemy wspomagania decyzji,* [dostęp: 15 sierpnia 2009] Dostępny w Internecie: http://ptf.fuw.edu.pl/fens/studia/ekonofizyka/olaf.morawski.pdf.;
17. Wrembel R., Królikowski Z., Morzy M., *Magazyny danych - stan obecny i kierunki rozwoju*. Poznań, ProDialog, 2000. [dostęp: 10 czerwca 2009] Dostępny w Internecie: http://www.cs.put.poznan.pl/mmorzy/papers/prodialog01.pdf.;
18. Mirończuk M., *System gromadzenia i przetwarzania medycznych danych pomiarowych w klinice urologii.* Politechnika Białystok, Białystok, 2007;
19. Staniszkis W., *Architektura Systemu Zarządzania wiedzą,* Studia i materiały Polskiego Stowarzyszenia Zarzadzania Wiedzą, 2005. s. 150 - 158;
20. Mirończuk M., *Eksploracja Danych w kontekscie procesu Knowledge Discovery In Databases (KDD) i metodologii Cross-Industry Standard Process for Data Mining (CRISP-DM),* Metody Informatyki Stosowanej, nr 2, 2009;
21. Paprzycki M., *Agenci programowi jako metodologia tworzenia oprogramowania,* [dostęp: 31 maja 2010] Dostępny w Internecie: http://www.e-informatyka.pl/wiki/Agenci_programowi_jako_metodologia_tworzenia_oprogramowania.;
22. Case S., Azarmi N., Thint M., Ohtani T., *Enhancing E-Communities with Agent-Based Systems,* Computer, nr. 34 2001;
23. Chandra J. M., Siddhartha M., *Multi agent based approach for analyzing spatial image data,* Page Help Intelligent Agent & Multi-Agent Systems, 2009 IAMA 2009, 2009;

24. Lieberman H., Rozenweig E., Singh P., *Aria: An Agent for Annotating and Retrieving Images,* Computer, nr. 34, 2001;
25. Wang J., Chen Y., *Agent-Based Price Negotiation System for Electronic Commerce,* Proceedings of the Seventh International Conference on Intelligent Systems Design and Applications, 2007;
26. Laird J. E., *Using a Computer Game to Develop Advanced AI,* Computer, nr. 37, 2001, s. 70-75;
27. Codd E. F., *A relational model of data for large shared data banks,* Communication of the ACM, nr. 13, 1970;
28. Connolly T., Begg C., *Systemy Baz Danych - projektowanie, wdrażanie i zarządzanie w praktyce.* Warszawa, 2003;
29. McJones P., *The 1995 SQL Reunion: People, Projects, and Politics,* 1995. [dostęp: 24 listopada 2009] Dostępny w Internecie: http://www.mcjones.org/System_R/SQL_Reunion_95/SRC-1997-018.pdf.;
30. Stones R., Matthew N., *Bazy danych i MySQL,* Od podstaw, 2003;
31. Meyer B., *Programowanie zorientowane obiektowo* 2005;
32. db4o, *Industry Solutions,* [dostęp: 19 listopad 2009] Dostępny w Internecie: http://www.db4o.com/about/solutions/;
33. *ObjectivityDB,* [dostęp: 19 listopad 2009] Dostępny w Internecie: http://www.objectivity.com/solutions/;
34. Golab L., Özsu M. T., *Issues in data stream management,* ACM SIGMOD Record, nr. 32, 2003, s. 5-14;
35. Babcock B., Babu S., Datar M., Motwani R., Widom J., *Models and Issues in Data Stream Systems.* Symposium on Principles of Database Systems Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2002;
36. Arasu A., Widom: J., A *Denotational Semantics for Continuous Queries over Streams and Relations*. ACM Sigmod Record, nr. 33, 2004, s. 6-11;
37. Maier D., Tufte K., Fernández-Moctezuma R. J., Tucker P., Li J., Papadimos V., *Niagara*. [dostęp: 20 listopad 2009] Dostępny w Internecie: http://datalab.cs.pdx.edu/niagara/;
38. *Stanford stream data manager*. [dostęp: 20 listopad 2009] Dostępny w Internecie: http://infolab.stanford.edu/stream/;
39. *Telegraph.* [dostęp: 20 listopad 2009] Dostępny w Internecie: http://telegraph.cs.berkeley.edu/index.html.;
40. Liu L., Pu C., Buttler D., Han W., Paques H., Tang W., *The Continual Queries* Dostępny w Internecie: [dostęp: 20 listopad 2009] http://www.cc.gatech.edu/projects/disl/CQ/;
41. Fisher K., Hogstedt K., Rogers A., Smith F., *Hancock*. [dostęp: 19 listopada 2009] Dostępny w Internecie: http://www2.research.att.com/~kfisher/hancock/;
42. Balakrishnan H., Stonebraker M., *Medusa*. [dostęp: 20 listopad 2009] Dostępny w Internecie: http://nms.lcs.mit.edu/projects/medusa/#people;
43. Berg B., *The Aurora Project.* [dostęp: 20 listopad 2009] Dostępny w Internecie: http://www.cs.brown.edu/research/aurora/;
44. Cattell R. G. G., *Object Data Management, Object-Oriented and Extended Relational Database Systems. Object Data Management, Object-Oriented and Extended Relational Database Systems,* Addison-Wesley, 1991;
45. *IBM. Introduction to OQL*. [dostęp: 20 listopad 2009] Dostępny w Internecie: http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/index.jsp?topic=/com.ibm.netcool_precision.doc/pr35se/xF1118340.html.;

46. Giero M., Nilewski R., *Wykorzystanie BTBD do zapisywania i rzeszukiwania danych o leczeniu niepłodoności metodami IVF*. [Hołny Majera]: Technologie Eksploracji i Reprezentacji Wiedzy - Hołny Majera 2008, 2008;
47. Praca zbiorowa, *Temporal Database*. [dostęp: 20 listopad 2009] Dostępny w Internecie: http://en.wikipedia.org/wiki/Temporal_database;
48. Snodgrass R. T., *The TSQL2 Temporal Query Language. Computer Science Department of the University of Arizona*, Retrieved 14 July 2009;
49. *TimeDB*. [dostęp: 19 listopad 2009] Dostępny w Internecie: http://www.timeconsult.com/Software/Software.html.;
50. Sellis T., Frank A. U., Grumbach S., Güting R. H., Jensen C. S., et al., *Spatio-Temporal Databases The CHOROCHRONOS Approach*. 2003. s. 297-307;
51. Vries A. P. D., *Content and multimediadatabase management systems. Centre for Telematics and Information Technology*, University of Twente, 1999;
52. Ghafoor A., *Multimedia database management systems. ACM Computing Surveys (CSUR)*, nr. 27, 1995;
53. Kozielski S., Małysiak B., Kasprowski O., Mrozek D., *Bazy danych: Struktury, Algorytmy, Metody,* WKŁ, 2006;
54. Nicola M., Linden B., *Native XML Support in DB2 Universal Database,* Very Large Data Bases archive Proceedings of the 31st international conference on Very large data bases, 2005;
55. Georgiadis H., Vassalos V., *Xpath on steroids: exploiting relational engines for xpath performance,* International Conference on Management of Data archive Proceedings of the 2007 ACM SIGMOD international conference on Management of data, 2007;
56. Morzy M., *Aktywne hurtownie danych, [Warszawa]: Najnowsze rozwiązania i praktyczne zastosowania: hurtownie danych i business intelligence,* Centrum Promocji Informatyki, 2004. [dostęp: 23 March 2004];
57. Traczyk T., *Hurtownie danych – wprowadzenie*. Infofestiwal'98, 1998;
58. Niederliński A., *Regułowo-Modelowe Systemy Ekspertowe,* Gliwice, PKJS, 2006;
59. Praca zbiorowa, *Deductive database,* [dostęp: 17 grudnia 2009] Dostępny w Internecie: http://en.wikipedia.org/wiki/Deductive_database;
60. Savinov A., *Principles of the Concept-Oriented Data Model,* 2004. [dostęp: 22 grudnia 2009] Dostępny w Internecie: http://conceptoriented.com/savinov/publicat/imi-report'04.pdf.
61. Savinov A. A., *Concept-Oriented Model and Query Language,* CoRR, nr. abs/0901.2224, 2009;
62. Savinov A., *Informal introduction into the Concept-Oriented Data Model,* 2005. [dostęp: 22 grudnia 2009] Dostępny w Internecie: http://conceptoriented.org/papers/ComInformalIntroduction.pdf.;
63. Savinov A., *Concept-Oriented Model. In: Ferraggine V.E., Doorn J.H. and Rivero L.C*., editors. Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends: IGI Global, 2009;
64. Wille R., Franzke C., *Formal Concept Analysis: Mathematical Foundations*, Springer Verlag, 1999;
65. Wolff K. E., *A first course in formal concept analysis,* 1994. [dostęp: 22 grudnia 2009] Dostępny w Internecie: http://www.fbmn.fh-darmstadt.de/home/wolff/Publikationen/A_First_Course_in_Formal_Concept_Analysis.pdf.
66. Griffin J., *Architektura dwu-, trzy- i n-warstwowa. XML I SQL* Server 2000, 2002. s. 136-139;

67. Węgrzyn A., Małecki M., *Oracle JDeveloper Suite 2.0 jako wydajne środowisko do tworzenia aplikacji intra- i internetowych, na przykładzie sklepu elektronicznego,* [Zakopane]: V Konferencja PLOUG, 1999. [dostęp: Październik 1999] Dostępny w Internecie: http://www.ploug.org.pl/konf_99/pdf/10.pdf.
68. Sienkiewicz J. E., Syty P., *Architektura warstwowa aplikacji internetowych,* [dostęp: 10 listopad 2009] Dostępny w Internecie: http://www.mif.pg.gda.pl/homepages/sylas/articles/art_2008_elblag.pdf.;
69. Krasuski A., *Usługi katalogowe w architekturze rozproszonej bazy danych w procesie wspomagania decyzji w Państwowej Straży Pożarnej,* Seminarium Zakładu Logiki Matematycznej, 2009;
70. Praca zbiorowa, *OLTP*. [dostęp: 10 listopad 2009] Dostępny w Internecie: http://it-portal.pl/pl/baza-wiedzy/slownik-kategorii-it/246-online-transaction-processing-oltp.html.;
71. *OLAP Council White Paper,* [dostęp: 10 listopad 2009] Dostępny w Internecie: www.olapcouncil.org/research/whtpaply.htm.;
72. Wilk-Kołodziejczyk D., *Pozyskiwanie wiedzy w sieciach komputerowych z rozproszonych źródeł informacji*. [w:] Lesław H.H., editor. Społeczeństwo informacyjne Wizja czy rzeczywistość? [on-line] Kraków: Uczelniane Wydawnictwa Naukowo - Dydaktyczne, 2003, 30 maja. [dostęp: 16 listopada 2007] Dostępny w Internecie: http://winntbg.bg.agh.edu.pl/skrypty2/0095/285-295.pdf.;
73. Kryszkiewicz M., *Eksploracja danych w telekomunikacji,* III edycja konferencji HURTOWNIE DANYCH I BUSINESS INTELLIGENCE - problematyka, rozwiązania, zastosowania, 2004;
74. Romei A., Ruggieri S., Turini F., *KDDML: a middleware language and system for knowledge discovery in databases,* Data & Knowledge Engineering, 2006. s. 179-220;
75. KDDML, *Knowledge Discovery in Databases Markup Language;*
76. Paton N. W., Díaz O., *Active database systems,* ACM Computing Surveys (CSUR), nr. 31, 1999, s. 63-103;
77. Bailey J., Poulovassilis A., Wood P. T., *An event-condition-action language for XML,* International World Wide Web Conference archive Proceedings of the 11th international conference on World Wide Web, 2002;
78. Mirończuk M., Maciak T., *Hybrydowy system wspomagania decyzji w otoczeniu komponentów platformy planowania zasobów przedsiębiorstwa dla Państwowej Straży Pożarnej,* Zeszyty Naukowe SGSP, nr. 42, 2010 (preprint);
79. Mirończuk M., Maciak T., *Problematyka projektowania modelu hybrydowego systemu wspomagania decyzji dla Państwowej Straży Pożarnej,* Zeszyty Naukowe SGSP, nr. 39, 2009;
80. Mirończuk M., Maciak T., *Projektowanie hybrydowego systemu wspomagania decyzji dla służb ratowniczych PSP – wybrane problemy i ich rozwiązania,* Zeszyty Naukowe SGSP, nr. 42, 2010 (preprint);
81. Podlaski Urząd Wojewódzki w Białymstoku Urząd Marszałkowski Województwa Podlaskiego, *System Zarządzania Kryzysowego w woj. podlaskim*. Białystok, 2003;
82. Schlossnagle G., *RPC - współpraca ze zdalnymi usługami. PHP Zaawansowane programowanie Vademecum profesjonalisty,* Warszawa: Helion, 2004. s. 393;
83. Horstmann C. S., Cornell G., *Obiekty rozproszone. Core Java Techniki zaawansowane,* Warszawa: Helion, 2009. s. 851;
84. *OMG. CORBA BASICS*. [dostęp: 13 czerwca 2010] Dostępny w Internecie: http://www.omg.org/gettingstarted/corbafaq.htm.;
85. *JADE,* [dostęp: 13 czerwca 2010] Dostępny w Internecie: http://jade.tilab.com/;

86. Krasuski A., Maciak T., *Rozproszone bazy danych w Państwowej Straży Pożarnej − model systemu*. [w:] Kozielski T., editor. Bazy danych, technologie, narzędzia. Warszawa WKŁ, 2005. s. 135−142;
87. Krasuski A., *Rozproszona baza danych − możliwości wykorzystania w PSP,* Przegląd Pożarniczy, nr. 5, 2006, s. 30−33;
88. Rozporządzenie Ministra Spraw Wewnętrznych i Administracji z dnia 29 grudnia 1999 r. w sprawie szczegółowych zasad organizacji krajowego systemu ratowniczo-gaśniczego. Dz.U.99.111.1311 § 34 pkt. 5 i 6;
89. Regulamin Powiatowego Stanowiska Kierowania. [dostęp: 13 czerwca 2010] Dostępny w Internecie: http://www.kppspblonie.pl/psk_regulamin.htm.;
90. *BIP. Powiatowe Stanowisko Kierowania*. [dostęp: 13 czerwca 2010] Dostępny w Internecie: http://bip-kppsp.spsieradz.finn.pl/index.jsp?bipkod=/003/007.

Recenzenci:
prof. dr hab. inż. Tadeusz Maciak
mgr inż. Jerzy Maciak