POLITECHNIKA GDAŃSKA

**Wydział Elektroniki,
Telekomunikacji i Informatyki**

Tomasz Boiński

Procedury odwzorowania i łączenia ontologii dziedzinowych

Rozprawa doktorska

Promotor:
prof. dr hab. inż. Henryk Krawczyk
Wydział Elektroniki, Telekomunikacji
i Informatyki
Politechnika Gdańska

Gdańsk, 2012

Spis treści

Projekt został sfinansowany ze środków Narodowego Centrum Nauki w ramach grantu
N N516 476440

Podziękowania

Chciałbym serdecznie podziękować przede wszystkim Rodzicom za ich wsparcie w całym moim dotychczasowym życiu, a zwłaszcza za dane mi możliwości i przekazaną wiedzę. Dziękuję również mojej Żonie za wsparcie i motywację, tak konieczną by udało się ukończyć pracę nad niniejszą rozprawą.

W sposób szczególny dziękuję Profesorowi Henrykowi Krawczykowi za opiekę naukową, zwłaszcza w trakcie realizacji rozprawy, oraz dane mi możliwości i wsparcie w całej pracy zawodowej i naukowej.

Wykaz symboli i skrótów

$\|A\|$	moc zbioru A
$\mathcal{B}$	byt
$\mathcal{K}$	klasa
$\mathcal{O}$	ontologia
$\mathcal{R}$	relacja
$\top$	koncept uniwersalny
$\bot$	koncept pusty
$\neg, \sqcap, \sqcup, \sqsubseteq, \equiv$	negacja, przecięcie, suma, zawieranie oraz równość konceptów
$\exists$	kwantyfikator egzystencjalny
$\varphi(\cdot)$	odwzorowanie ontologii
$\oplus$	złączenie ontologii
$I(description(A, B))$	opis pojęć A i B
$I(common(A, B))$	wspólność znaczeniowa pojęć A i B
sim_{lin}	wartość podobieństwa wyznaczana miarą opracowaną przez Lin
sim_{lev}	wartość podobieństwa wyznaczana na podstawie odległości edycyjnej Levenshteina
P_{kom}	podobieństwo komentarzy przypisanych do klas i bytów
P_{lex}	podobieństwo leksykalne klas i bytów
P_{sem}	podobieństwo semantyczne klas i bytów
P_{str}	podobieństwo strukturalne klas i bytów
ENISA	European Network and Information Security Agency
NIST	National Institute of Standards and Technology
OCS	Ontology Creation System
SOVA	Simple Ontology Visualization API
SUMO	The Suggested Upper Merged Ontology

Rozdział 1

Wprowadzenie

1.1 Sieć WWW a Sieć Semantyczna

Wraz z rozwojem Internetu to, co stanowiło o jego sile – prostota języka HTML, staje się obecnie przeszkodą w jeszcze lepszym i efektywniejszym wykorzystaniu jego zasobów. Sieć WWW dopiero wtedy osiągnie swój prawdziwy potencjał, kiedy możliwe będzie udostępnianie i przetwarzanie informacji zarówno przez niezależne zautomatyzowane narzędzia, jak i ludzi. Dążąc do rozwiązania tego problemu Tim Berners-Lee zaproponował ideę Sieci Semantycznej [15, 43], w której zasoby są powiązane ze sobą znaczeniowo. Jest ona rozszerzeniem sieci WWW mającym na celu nadanie obecnej w niej informacji dobrze zdefiniowanego znaczenia, tak by rozumiały ją zarówno maszyny, jak i ludzie. Podejście to umożliwi swobodną wymianę informacji oraz formalizację i unifikację już zgromadzonej wiedzy. Dobrze określone i sformalizowane struktury umożliwią łatwiejsze i dokładniejsze wyszukiwanie informacji, czy wnioskowanie nowych, nie zapisanych explicite, faktów.

W tradycyjnym podejściu wynik zapytania zwrócony przez wyszukiwarkę internetową w dużej mierze zależy od poprawności zadanego pytania. Szukając np. pewnej osoby musielibyśmy podać jej nazwisko i sami przeanalizować wyniki w celu ich weryfikacji. Sieć Semantyczna umożliwi nam podanie wybranego szczegółowego faktu na temat poszukiwanej osoby. Na podstawie sformalizowanych skojarzeń wyszukiwarka będzie w stanie zwrócić kompletną listę osób pasujących do podanego faktu.

Zaproponowana wizja światowej pajęczyny Sieci Semantycznej wymaga do realizacji efektu „kuli śnieżnej", czyli powszechnego zastosowania technologii semantycznych w światowych zasobach Internetu. Standaryzacją Sieci Semantycznej zajmuje się W3C Semantic Web Activity [103]. Główne działania na tym polu ograniczają się jednak do określania języka jej zapisu. Brak jest niestety ustalonych i powszechnie akceptowanych standardów budowy sieci, takich jak np. nazewnictwo pojęć, czy związków pomiędzy nimi.

1.2 Ontologie a wiedza

W naszym codziennym życiu często spotykamy się z pojęciami *informacja* oraz *wiedza*. Pojęcia te często wykorzystywane są jako synonimy. W rzeczywistości jednak ich znaczenie jest diametralnie różne.

Informacja (łac. *informatio* – przedstawienie, wizerunek; *informare* – kształtować, przedstawiać) jest terminem interdyscyplinarnym, definiowanym różnie w różnych dziedzinach nauki. Najogólniej, cytując Cackowskiego i innych [34] „informacja jest to właściwość pewnych obiektów". Jest więc pewnym swoistym opisem precyzujący jakiś fragment rzeczywistości.

Wiedza z kolei jest to „ogół wiarygodnych informacji o rzeczywistości wraz z umiejętnością ich

wykorzystywania" [84]. Jest więc usystematyzowaną i sprawdzoną informacją, którą potrafimy w jakiś sposób wykorzystać.

Komputery pozwalają człowiekowi gromadzić zarówno wiedzę, jak i informacje. Same, przez fakt braku świadomości, mogą operować jedynie na informacji. Nie są w stanie jednak określić jej poprawności ani uzależnić jej znaczenia od kontekstu, czy własnych doświadczeń. Nie mogą więc przeprowadzać operacji nie podlegających algorytmizacji. Podjęto jednak próby dostosowania wiedzy do możliwości maszyn. Poprzez formalizację metod zapisu informacji, wzbogacenie jej o kontekst i dobrze określone związki pomiędzy poszczególnymi jej fragmentami możliwe stało się realizowanie coraz to bardziej złożonych zadań.

Jednym z takich formalnych zapisów wiedzy są ontologie. Ich wszechstronność, poziom formalizmu i bogactwo wyrazu okazały się nieocenionym sprzymierzeńcem w procesie rozszerzania możliwości poznawczych i analitycznych komputerów.

1.3 Ontologia jako element Sieci Semantycznej

Kluczowym elementem w konstrukcji globalnej semantycznej pajęczyny jest pojęcie ontologii. Ontologia jest konceptem interdyscyplinarnym i jako takie zyskało wiele znaczeń. Wywodzi się z języka greckiego i oznacza naukę o tym co jest. W literaturze filozoficznej termin „ontologia" pojawił się w XVII wieku. Po raz pierwszy został użyty przez Rudolfa Gocleniusa w słowniku filozoficznym pt. *Lexicon philosophicum quo tamquam clavae philosophiae fors aperiuntur* [64]. Termin „ontologia" nie figurował w nim jako niezależne hasło, lecz jako grekojęzyczny komentarz przy haśle abstrakcja. Po raz pierwszy słowa „ontologia" jako niezależnego hasła użył filozof i teolog Johannes Micraelius, a następnie w swych dziełach również filozof i teolog Johannes Clauberg. Termin spopularyzowany został przez Christiana Wolffa poprzez podział filozofii na ontologię, kosmologię i psychologię [133].

Również w informatyce ontologie są konstrukcją nie posiadającą jednej, precyzyjnej definicji. Znaczenie tego słowa zmienia się w zależności od podejścia czy stopnia sformalizowania. Najpopularniejszą definicją jest podana przez Grubera w 1993 roku: „Ontologia jest bezpośrednią specyfikacją konceptualizacji" [70], czyli pewnym abstrakcyjnym modelem opisującym wybrany fragment świata (konceptualizacja) opisany jednoznacznym, formalnym językiem umożliwiającym zrozumienie i interpretację tego modelu (bezpośrednia specyfikacja). Definicja ta pozostawia jednak szerokie pole interpretacji co do wymaganego sposobu sformalizowania zapisów, poziomu szczegółowości, czy złożoności ontologii, umożliwia jednak zawężenie pola interpretacji tego pojęcia do reprezentowania wiedzy poprzez zdefiniowanie zbioru pojęć i relacji miedzy nimi. Ontologia jest więc swoistym opisem dla danych. Siła sieci Semantycznej polega na fakcie, że dowolny podmiot może stworzyć ontologię na dowolny temat [55, 141] przyczyniając się tym samym do rozwoju światowej pajęczyny. Formalizm zawarty w ontologii pozwoli komputerom na samodzielne przetwarzanie informacji bez udziału człowieka czy wspieranie go w procesie decyzyjnym poprzez sugestię rozwiązań, czy kierunku analizy problemu. Z tego punktu widzenia ontologie wraz z Siecią Semantyczną postrzegane są jako rozwiązanie dla heterogeniczności danych w Internecie. Wraz ze wzrostem zainteresowaniem tym tematem, w różnych dziedzinach pojawiały się coraz to nowe ontologie. Niestety brak standardów wytwarzania ontologii spowodował ich olbrzymią różnorodność. Co prawda powstał standard reprezentacji ontologii w postaci Resource Description Framwework (RDF) [30] czy wywodzący się z niego język OWL Web Ontology Language [3, 45, 102] jednak w samej strukturze czy nazewnictwie obiektów pozostaje pełna dowolność. W wyniku występowania mnogości podmiotów zajmujących się budowaniem Sieci Semantycznej same ontologie wprowadzają więc heterogeniczność. Powstaje problem, jak radzić sobie z tą nową różnorodnością. Zachodzi konieczność integracji wiedzy zawartej w poszczególnych ontologiach, tak by efekt był zrozumiały dla wszystkich korzystających z tych ontologii systemów.

Aby integracja była możliwa konieczne, są do spełnienia dwa warunki. By możliwa była integracja wiedzy, jej łączone fragmenty powinny być do siebie w jakimś sensie podobne. Aspekt

podobieństwa ontologii szerzej omówiony zostanie w dalszej części rozprawy. Na wstępie sprecyzować możemy jednak pewną intuicyjną myśl kryjącą się za zwrotem „ontologie dziedzinowe" zawartym w tytule niniejszej rozprawy. Przy integracji ontologii, których konstrukcja w dużej mierze oparta jest o pojęcia opisane za pomocą słów w języku naturalnym, niezbędna jest możliwość ustalenia kontekstu określającego znaczenie użytych słów. W przeciwnym wypadku nie będziemy w stanie odróżnić np. pojęcia klucz oznaczającego klucz szyfrujący algorytmu kryptograficznego od pojęcia klucz oznaczającego przedmiot otwierający zamek w drzwiach. Ontologie muszą więc opisywać wspólny fragment rzeczywistości, czyli operować na pewnej dziedzinie tematycznej. W jej obrębie zakres informacji, jej szczegółowość, czy prezentowany punkt widzenia, mogą być zgoła odmienne. Im większe podobieństwo integrowanych ontologii, tym większe prawdopodobieństwo, że operacje ich łączenia lub odwzorowywania przebiegną pomyślnie, a wynikowa ontologia będzie czymś więcej niż tylko połączeniem obu ontologii za pomocą dodatkowego węzła.

Drugim aspektem są odpowiednie narzędzia i algorytmy umożliwiające przeprowadzenie integracji poprzez wykrywanie części wspólnych pomiędzy łączonymi ontologii, czy określania zależności pomiędzy wchodzącymi w ich skład elementami. W niniejszej rozprawie podjęta została próba opracowania takich algorytmów i wskazania wspierających je narzędzi gwarantujących poprawność ich wykonania i ocenę ich przydatności w procesie integracji wiedzy.

1.4 Cel i tezy pracy

Mnogość podmiotów zajmująca się tworzeniem ontologii oraz różnorodność wypracowanych przez nie rozwiązań prowadzi do następujących celów pracy:

- Analiza algorytmów łączenia oraz odwzorowywania ontologii dziedzinowych.
- Opracowanie procedur automatycznego przeprowadzania tych operacji.

Osiągnięcie w.w. celów pozwoli na udowodnienie tez pracy:

1. Problem integracji systemów informatycznych wymaga odwzorowywania i łączenia ontologii opisujących obszary działania tych systemów.
2. Algorytm łączenia i odwzorowywania ontologii może zostać skonstruowany w oparciu na analizie leksykalnej, semantycznej i strukturalnej elementów tych ontologii, przy zachowaniu wielomianowej złożoności obliczeniowej.
3. Zaproponowany algorytm jest realizowalny i możliwy do wykorzystania dla ontologii dziedzinowych dotyczących różnych dziedzin zastosowań, a w szczególności bezpieczeństwa systemów komputerowych.

1.5 Układ pracy

Podział dalszej części pracy jest następujący. Rozdział drugi definiuje pojęcie ontologii oraz ich integracji, a także prezentuje podstawowe zagadnienia z dziedziny inżynierii ontologii mające wpływ na proces integracji wiedzy. Rozdział trzeci przybliża istniejące techniki odwzorowywania i łączenia ontologii dziedzinowych oraz algorytmy i narzędzia pomocne w procesie określania podobieństwa pomiędzy pojęciami. W rozdziale czwartym zaproponowano algorytm integracji ontologii oraz zestaw wspierających go algorytmów pomocniczych określających podobieństwo pomiędzy konceptami wchodzącymi w skład integrowanych ontologii. W rozdziale piątym zaprezentowano system OCS będący narzędziem wspierającym proces tworzenia, integracji oraz składowania ontologii. Rozdział szósty prezentuje analizę jakości algorytmu oraz prezentuje wyniki testów. Rozdział siódmy stanowi podsumowanie rozprawy.

Rozdział 2

Ontologia jako reprezentacja świata rzeczywistego

2.1 Wprowadzenie

W rozdziale sformułowano definicję ontologii oraz zaproponowano język OWL jako formę jej reprezentacji. Przyjęto również graficzną reprezentację ontologii w postaci grafu, którego wierzchołkami są pojęcia i właściwości zawarte w ontologii, a krawędziami występujące pomiędzy nimi relacje. Określono także czym jest dziedzinowość ontologii i sklasyfikowano formy heterogeniczności w spojrzeniu na tematykę ontologii w ujęciu komputerowego przetwarzania wiedzy.

W dalszej części rozdziału zdefiniowano pojęcia odwzorowania oraz łączenia ontologii, a także określono elementy składowe procedur łączenia i odwzorowania ontologii dziedzinowych.

W kolejnym rozdziale przedstawiono przegląd istniejących technik odwzorowania oraz łączenia ontologii dziedzinowych.

2.2 Ogólna definicja ontologii

Jak już wspomniano we wprowadzeniu do niniejszej rozprawy nie ma jednej ustalonej definicji ontologii. Ta wprowadzona przez Grubera jest na tyle ogólna, że pozostawia szerokie pole interpretacji. W niniejszej pracy proponowane jest przyjęcie definicji ontologii wynikającej z połączenia podejścia Hovy'ego [75] oraz Euzenata i Volcheva [53]. Ontologia rozumiana jest jako zbiór klas i bytów powiązanych ze sobą relacjami dziedziczenia i przynależności. W skład ontologii wchodzą zatem następujące elementy:

- klasa – jest to struktura agregująca elementy świata rzeczywistego, posiadający wspólny zbiór atrybutów (ale o nieokreślonej wartości) zdefiniowanych w tej klasie, np. klasa algorytmów szyfrujących opartych na algorytmie RSA. Przyjęto również występowanie specjalnej klasy „Coś" T będącej klasą, po której dziedziczą wszystkie inne klasy, oraz do której przynależą wszystkie byty w ontologii. Dzięki temu analizowany graf jest zawsze grafem spójnym. Klasa reprezentuje więc pewne *pojęcie* (inaczej *koncept*, ang. *concept*) opisujące spójny fragment rzeczywistego świata,

- byt – reprezentuje konkretny element świata rzeczywistego posiadający skonkretyzowane wartości parametrów, np. algorytm szyfrujący oparty na algorytmie RSA posiadającym klucze o długości 1024 bity,

- komentarz – jest to dodatkowy opis słowny przypisany do klasy lub bytu niosący dodatkową informację o znaczeniu danego konceptu,
- relacje – są to wszelkiego rodzaju związki pomiędzy klasami i bytami reprezentujące ich wzajemne zależności. Podstawowymi relacjami są:
  - relacja dziedziczenia – relacja ta może zachodzić pomiędzy dwiema klasami, umożliwia agregacje klas w bardziej ogólne koncepty. Mówimy, że klasa X dziedziczy po klasie Y, jeżeli każdy element klasy X jest również elementem klasy Y. Klasa dziedzicząca posiada wszystkie właściwości zdefiniowane w klasie dziedziczonej, jest to też relacja przechodnia, np. klasa algorytmów szyfrujących opartych na algorytmie RSA dziedziczy po klasie algorytmów szyfrujących asymetrycznych,
  - relacja przynależności – relacja ta może zachodzić pomiędzy bytem a klasą, umożliwia powiązanie bytu z klasą definiująca właściwości tego bytu, np. algorytm szyfrujący oparty na algorytmie RSA posiadającym klucze o długości 1024 bity przynależy do klasy algorytmów szyfrujących opartych na algorytmie RSA.

W celu zachowania siły wyrazu wprowadzono dodatkowe relacje:

- tożsamości – oznaczającą, że dwie klasy lub dwa byty są identyczne,
- rozłączności – oznaczającą, że dwie klasy lub dwa byty są różne,
- unii – relacja zachodząca tylko pomiędzy klasami, mówiąca, że klasa będąca wynikiem unii jest złączeniem klas wchodzących w jej skład, czyli element x należy do unii klas jeżeli należy do przynajmniej jednej z klas tworzących unię,
- przecięcia – relacja zachodząca tylko pomiędzy klasami, mówiąca, że klasa będąca wynikiem przecięcia jest częścią wspólną klas wchodzących w jej skład, czyli element x należy do przecięcia klas jeżeli należy do każdej z klas tworzących unię.

Na potrzeby niniejszej rozprawy, ontologię $\mathcal{O}$ możemy więc zdefiniować jako krotkę:

$$\mathcal{O}(\mathcal{K}, \mathcal{B}, \mathcal{C}, \mathcal{R}) \tag{2.1}$$

gdzie:
$\mathcal{K}$ - zbiór klas wchodzących w skład ontologii, wraz z wyróżnionym konceptem uniwersalnym $\top$ oraz konceptem pustym $\bot$
$\mathcal{B}$ - zbiór bytów wchodzących w skład ontologii,
$\mathcal{C}$ - zbiór komentarzy przypisanych do klas $\mathcal{K}$ i bytów $\mathcal{B}$ wchodzących w skład ontologii,
$\mathcal{R}$ - zbiór relacji wiążących elementy ontologii.

Relacje $\mathcal{R}$ z kolei można zdefiniować jako krotkę:

$$\mathcal{R}(\mathcal{D}, \mathcal{P}, \mathcal{T}, \mathcal{S}, \mathcal{U}, \mathcal{I}) \tag{2.2}$$

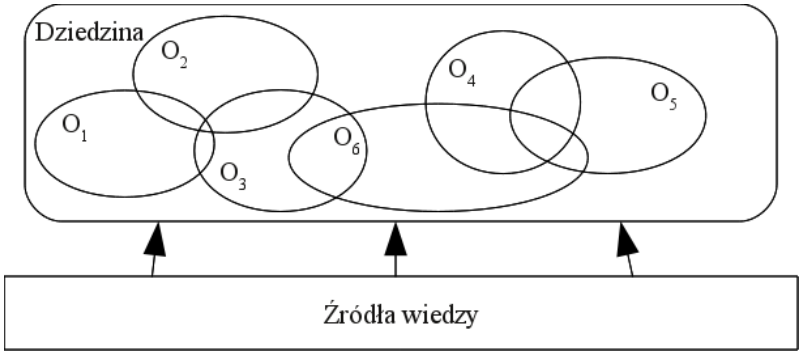
gdzie:
$\mathcal{D}$ - zbiór relacji dziedziczenia wchodzących w skład ontologii,
$\mathcal{P}$ - zbiór relacji przynależności wchodzących w skład ontologii,
$\mathcal{T}$ - zbiór relacji tożsamości wchodzących w skład ontologii,
$\mathcal{S}$ - zbiór relacji rozłączności wchodzących w skład ontologii,
$\mathcal{U}$ - zbiór relacji unii wchodzących w skład ontologii,
$\mathcal{I}$ - zbiór relacji przecięcia wchodzących w skład ontologii.

Jako język zapisu ontologii zgodnych z powyższym modelem zaproponowano język OWL [45], będącym najpopularniejszym obecnie językiem zapisu ontologii, a zawarte w nich dane tekstowe (nazwy klas i bytów, komentarze) będą w języku angielskim.

Jako reprezentację graficzną ontologii przyjęto graf skierowany o dwu rodzajach wierzchołków. Jeden rodzaj wierzchołka (owal wypełniony kolorem niebieskim) reprezentuje klasy a drugi (prostokąt wypełniony kolorem szarym) byty. Klasy będące wynikiem relacji unii i przecięcia oznaczone są okręgiem wypełnionym kolorem żółtym i wpisanym symbolem odpowiednio $\bigcup$ oraz $\bigcap$. Relacje zachodzące pomiędzy klasami oraz bytami i klasami stanowią krawędzie grafu. Relacja dziedziczenia oznaczona jest strzałką o grocie zamkniętym, niewypełnionym, skierowanym od klasy dziedziczącej do klasy dziedziczonej. Relacja przynależności oznaczona jest linią bez grotu (kierunek jest tutaj domyślnie przyjęty jako zwrócony od bytu do klasy). Relacja tożsamości oznaczona jest linią o dwu grotach otwartych skierowanych do klas tożsamych. Relacja rozłączności oznaczona jest podobnie jak relacja tożsamości, przy czym groty skierowane są od klas. Relacje przynależności klas do unii i przecięcia oznaczone są strzałkami o grotach okrągłych, pustych zwróconych w kierunku symbolu unii/przecięcia. Szczegółową notację opisującą wszystkie elementy graficznej reprezentacji ontologii umieszczono w sekcji 5.8 na stronie 68.

2.3 Ontologie dziedzinowe

Każdy człowiek charakteryzuje się indywidualnym spojrzeniem na otaczającą go rzeczywistość. Spojrzenie to może być mniej lub bardziej podobne do poglądów innych osób czy ogólnie przyjętego konsensusu. Dla dowolnie szerokiej dziedziny możliwe jest więc utworzenie różnych ontologii w odmienny sposób opisujących daną dziedzinę (rys. 2.1).



Rysunek 2.1: Dziedzina i utworzone w niej ontologie

Różnorodność poglądów wprowadza więc heterogeniczność rozwiązań. Form heterogeniczności może być bardzo dużo [28], jednak z punktu widzenia komputerowego przetwarzania wiedzy istotne są następujące kategorie:

- składniowa – wynika z wyboru języka zapisu ontologii. Obecnie dostępnych jest wiele języków, takich jak OWL, RDF, PROLOG, KIF itp. Każdy z nich ma sobie właściwą składnię i różną siłę wyrazu czy stopień złożoności. Języki te są jednak dobrze opisane, a problem konwersji między nimi rozpoznany,
- leksykalna – wynika z bogactwa języka naturalnego, który jest używany do nazywania elementów składowych ontologii. W dowolnym języku naturalnym występują słowa czy zwroty, których znaczenie zmienia się w zależności od kontekstu. Przykładem może być, wspomniane w wstępie do niniejszej rozprawy, słowo „klucz", które może opisywać różne koncepty w zależności od kontekstu, czy słowa „samochód" oraz „auto", które pomimo innego brzmienia i pisowni mają identyczne znaczenie,
- poglądowa – wynika z odmienności spojrzenia na świat twórcy ontologii. Podążając za [28] możemy wyróżnić tutaj kolejne trzy kategorie:
  - pokrycie – ontologie mogą odnosić się do różnych fragmentów dziedziny, pokrywających się całkowicie, częściowo lub wcale,

– ziarnistość – ontologie mogą różnić się między sobą szczegółowością opisu czy poziomem abstrakcji,

– perspektywa – ontologie mogą reprezentować różny punkt widzenia nawet przy zachowaniu identycznego pokrycia czy ziarnistości.

Te trzy kategorie mogą w dowolny sposób przenikać się tworząc dowolną strukturę różnorodności poglądowej.

Interoperacyjność wiedzy wymaga pokonania każdej z powyższych trudności. Ze względu na dobre rozpoznanie problemu w literaturze w niniejszej rozprawie nie będziemy dalej zajmować się różnorodnością składniową. Przyjęto założenie, że integrowane ontologie zapisane są w jednym, wspólnym języku OWL, jednak łatwość transformacji jednej reprezentacji w drugą nie pomniejsza ogólności proponowanego rozwiązania. Wyzwanie stanowią różnice leksykalne i poglądowe. Wymagają one od systemu komputerowego realizującego integrację wiedzy zdolności do zarówno konstrukcji nowej hierarchii, jak i dopasowania leksykalnego etykiet konceptów. W dalszych rozdziałach niniejszej rozprawy opisana zostanie propozycja rozwiązania tych problemów.

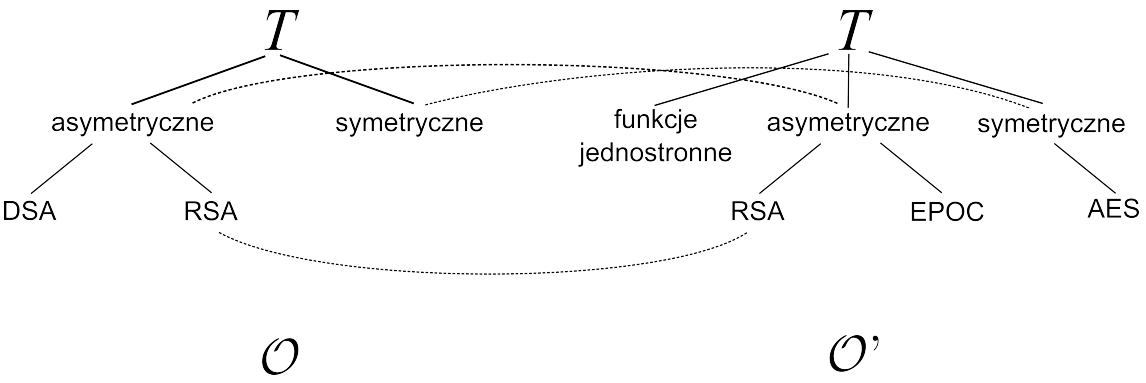
2.4 Czym jest integracja ontologii?

Mnogość źródeł, a tym samym rozproszenie wiedzy, wymaga sprawnych mechanizmów wyszukiwania i pozyskiwania informacji. Wiedza zawarta w różnych ontologiach musi być najpierw scalona, by móc skorzystać z niej jako całości. Proces scalania wiedzy zawartej w ontologii wymaga zdolności do scalania, czyli integrowania ontologii. W tym celu możliwe są dwa podejścia. Pierwsze to tworzenie odwzorowań jednych ontologii w inne, czyli tworzenie powiązań między klasami, rolami oraz relacjami integrowanych ontologii bez ingerencji w ich konstrukcję. Drugie to łączenie ontologii , w wyniku którego powstaje nowa ontologia, zawierająca wszystkie elementy łączonych opisów.

Odwzorowaniem φ ontologii $\mathcal{O}$ w ontologię $\mathcal{O}'$ nazywamy funkcję 2.3 przekształcającą zbiór elementów wchodzących w skład ontologii $\mathcal{O}$ w zbiór elementów ontologii $\mathcal{O}'$.

$$\varphi(\mathcal{O}) = \mathcal{O}' \tag{2.3}$$

Funkcja φ tworzy więc zbiór powiązań pomiędzy elementami ontologii $\mathcal{O}$ i $\mathcal{O}'$ (rys. 2.2).



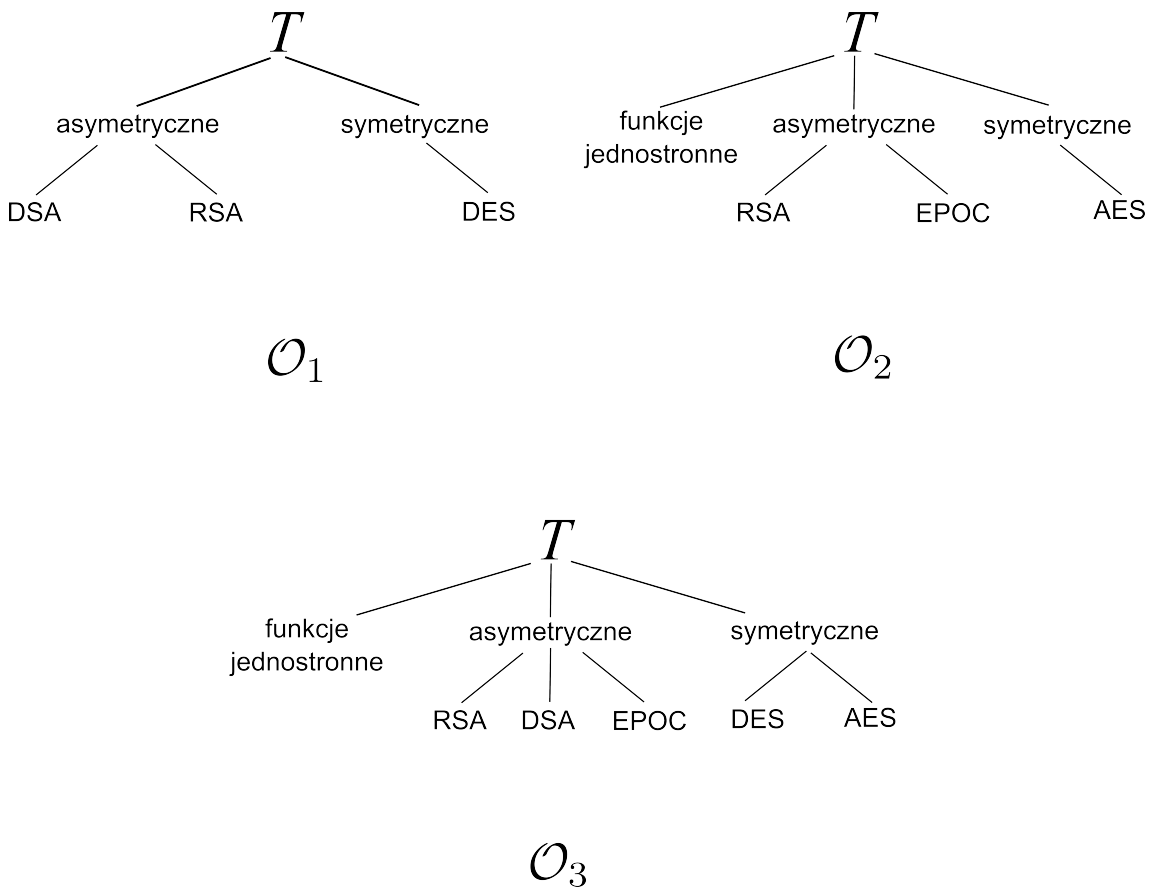
Rysunek 2.2: Odwzorowanie ontologii. Powiązania pomiędzy elementami oznaczone są linią przerywaną.

W języku OWL odwzorowanie można zaprezentować poprzez dyrektywę „owl:imports". Wydruk 2.1 prezentuje ontologię $\mathcal{O}$ a wydruk 2.2 ontologię $\mathcal{O}'$. Wydruk 2.3 prezentuje z kolei ontologię w języku OWL integrującą ontologię $\mathcal{O}$ oraz $\mathcal{O}'$ poprzez powiązania ich elementów.

Łączeniem $\oplus$ ontologii $\mathcal{O}_1$ oraz $\mathcal{O}_2$ nazywamy operację 2.4, w wyniku której powstaje nowa ontologia $\mathcal{O}_3$.

$$\mathcal{O}_3 = \mathcal{O}_1 \oplus \mathcal{O}_2 \tag{2.4}$$

W wyniku tej operacji tworzony jest więc całkowicie nowa ontologia, integrująca wiedzę obu łączonych ontologii, a nie tylko sam zbiór powiązań pomiędzy jej elementami(rys. 2.3).



Rysunek 2.3: Łączenie ontologii.

W języku OWL wynik łączenia ontologii $\mathcal{O}_1$, zaprezentowanej na wydruku 2.1, oraz ontologii $\mathcal{O}_2$, zaprezentowanej na wydruku 2.2, przyjmie postać nowej ontologii $\mathcal{O}_3$, zaprezentowanej na wydruku 2.4.

Należy zauważyć, że operacja łączenia może być przeprowadzona bezpośrednio na integrowanych ontologiach, jak i za pośrednictwem funkcji odwzorowującej ich elementy. Wybór drogi integracji wiedzy w dużej mierze uzależniony jest od potrzeb odbiorcy wyniku integracji.

Wydruk 2.1: Reprezentacja w języku OWL ontologii $\mathcal{O}$ oraz $\mathcal{O}_1$

```
<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>

<rdf:RDF xmlns="http://kask.eti.pg.gda.pl/securityA.owl#"
  xml:base="http://kask.eti.pg.gda.pl/securityA.owl"
```

```
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <owl:Ontology rdf:about="http://kask.eti.pg.gda.pl/securityA.owl"/>

 <!--
 ////////////////////////////////////////////////////////////////////////////////
 //
 // Classes
 //
 ////////////////////////////////////////////////////////////////////////////////
 -->

 <!-- http://kask.eti.pg.gda.pl/securityA.owl#DSA -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityA.owl#DSA">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityA.owl#asymetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityA.owl#RSA -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityA.owl#RSA">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityA.owl#asymetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityA.owl#asymetryczne -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityA.owl#asymetryczne"/>


 <!-- http://kask.eti.pg.gda.pl/securityA.owl#symetryczne -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityA.owl#symetryczne"/>
</rdf:RDF>
```

Wydruk 2.2: Reprezentacja w języku OWL ontologii $\mathcal{O}'$ oraz $\mathcal{O}_2$

```
<?xml version="1.0"?>

<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY securityB "http://kask.eti.pg.gda.pl/securityB.owl#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>

<rdf:RDF xmlns="http://kask.eti.pg.gda.pl/securityB.owl#"
   xml:base="http://kask.eti.pg.gda.pl/securityB.owl"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:securityB="http://kask.eti.pg.gda.pl/securityB.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <owl:Ontology rdf:about="http://kask.eti.pg.gda.pl/securityB.owl"/>

 <!--
 ////////////////////////////////////////////////////////////////////////////////
 //
 // Classes
 //
 ////////////////////////////////////////////////////////////////////////////////
 -->

 <!-- http://kask.eti.pg.gda.pl/securityB.owl#AES -->

 <owl:Class rdf:about="&securityB;AES">
  <rdfs:subClassOf rdf:resource="&securityB;symetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#EPOC -->

 <owl:Class rdf:about="&securityB;EPOC">
  <rdfs:subClassOf rdf:resource="&securityB;asymetryczne"/>
 </owl:Class>
```

```
 <!-- http://kask.eti.pg.gda.pl/securityB.owl#RSA -->

 <owl:Class rdf:about="&securityB;RSA">
  <rdfs:subClassOf rdf:resource="&securityB;asymetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#asymetryczne -->

 <owl:Class rdf:about="&securityB;asymetryczne">
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#funkcje_jednostronne -->

 <owl:Class rdf:about="&securityB;funkcje_jednostronne"/>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#symetryczne -->

 <owl:Class rdf:about="&securityB;symetryczne">
  <rdfs:subClassOf rdf:resource="&owl;Thing"/>
 </owl:Class>
</rdf:RDF>
```

Wydruk 2.3: Reprezentacja w języku OWL odwzorowania ontologii $\mathcal{O}$ w ontologię $\mathcal{O}'$

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
 <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
 <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
 <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
 <!ENTITY securityB "http://kask.eti.pg.gda.pl/securityB.owl#" >
 <!ENTITY securityA "http://kask.eti.pg.gda.pl/securityA.owl#" >
 <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>

<rdf:RDF xmlns="http://kask.eti.pg.gda.pl/securityC0.owl#"
   xml:base="http://kask.eti.pg.gda.pl/securityC0.owl"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:securityB="http://kask.eti.pg.gda.pl/securityB.owl#"
   xmlns:securityA="http://kask.eti.pg.gda.pl/securityA.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <owl:Ontology rdf:about="http://kask.eti.pg.gda.pl/securityC0.owl">
  <owl:imports rdf:resource="http://kask.eti.pg.gda.pl/securityA.owl"/>
  <owl:imports rdf:resource="http://kask.eti.pg.gda.pl/securityB.owl"/>
 </owl:Ontology>

 <!--
 ///////////////////////////////////////////////////////////////////////////////////////
 //
 // Classes
 //
 ///////////////////////////////////////////////////////////////////////////////////////
 -->

 <!-- http://kask.eti.pg.gda.pl/securityA.owl#RSA -->

 <rdf:Description rdf:about="&securityA;RSA">
  <owl:equivalentClass rdf:resource="&securityB;RSA"/>
 </rdf:Description>


 <!-- http://kask.eti.pg.gda.pl/securityA.owl#asymetryczne -->

 <rdf:Description rdf:about="&securityA;asymetryczne">
  <owl:equivalentClass rdf:resource="&securityB;asymetryczne"/>
 </rdf:Description>


 <!-- http://kask.eti.pg.gda.pl/securityA.owl#symetryczne -->
```

```
 <rdf:Description rdf:about="&securityA;symetryczne">
  <owl:equivalentClass rdf:resource="&securityB;symetryczne"/>
 </rdf:Description>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#RSA -->

 <rdf:Description rdf:about="&securityB;RSA"/>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#asymetryczne -->

 <rdf:Description rdf:about="&securityB;asymetryczne"/>


 <!-- http://kask.eti.pg.gda.pl/securityB.owl#symetryczne -->

 <rdf:Description rdf:about="&securityB;symetryczne"/>
</rdf:RDF>
```

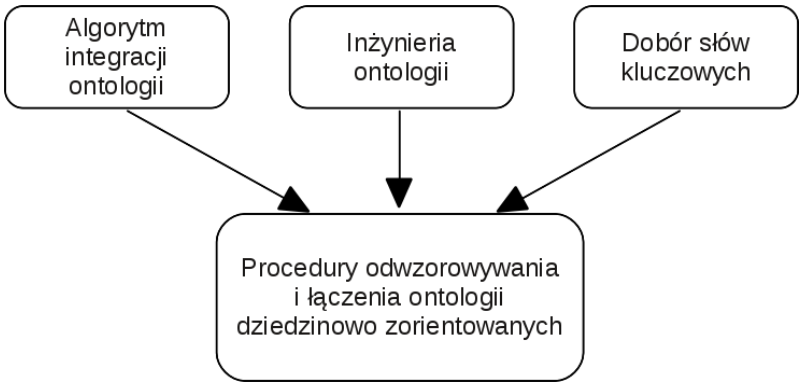
Wydruk 2.4: Reprezentacja w języku OWL ontologii $\mathcal{O}_3$ będącej złączeniem ontologii $\mathcal{O}_1$ oraz ontologii $\mathcal{O}_2$

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
 <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
 <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
 <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
 <!ENTITY securityB "http://kask.eti.pg.gda.pl/securityB.owl#" >
 <!ENTITY securityA "http://kask.eti.pg.gda.pl/securityA.owl#" >
 <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>

<rdf:RDF xmlns="http://kask.eti.pg.gda.pl/securityC.owl#"
   xml:base="http://kask.eti.pg.gda.pl/securityC.owl"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:securityB="http://kask.eti.pg.gda.pl/securityB.owl#"
   xmlns:securityA="http://kask.eti.pg.gda.pl/securityA.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
 <owl:Ontology rdf:about="http://kask.eti.pg.gda.pl/securityC.owl"/>

 <!--
 ///////////////////////////////////////////////////////////////////////////////////////
 //
 // Classes
 //
 ///////////////////////////////////////////////////////////////////////////////////////
 -->

 <!-- http://kask.eti.pg.gda.pl/securityC.owl#AES -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#AES">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityC.owl#symetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityC.owl#DSA -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#DSA">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityC.owl#asymetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityC.owl#EPOC -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#EPOC">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityC.owl#asymetryczne"/>
 </owl:Class>


 <!-- http://kask.eti.pg.gda.pl/securityC.owl#RSA -->

 <owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#RSA">
  <rdfs:subClassOf rdf:resource="http://kask.eti.pg.gda.pl/securityC.owl#asymetryczne"/>
 </owl:Class>
```

```
<!-- http://kask.eti.pg.gda.pl/securityC.owl#asymetryczne -->

<owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#asymetryczne">
 <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>


<!-- http://kask.eti.pg.gda.pl/securityC.owl#funkcje_jednostronne -->

<owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#funkcje_jednostronne"/>


<!-- http://kask.eti.pg.gda.pl/securityC.owl#symetryczne -->

<owl:Class rdf:about="http://kask.eti.pg.gda.pl/securityC.owl#symetryczne">
 <rdfs:subClassOf rdf:resource="&owl;Thing"/>
</owl:Class>
</rdf:RDF>
```

2.5 Procedury wykorzystywane przy odwzorowywaniu i łączeniu ontologii

Operacja łączenia i odwzorowania ontologii wykracza poza sam proces ich integracji. Sam algorytm wykonujący to zadanie jest tylko jedną z procedur niezbędnych do poprawności całego procesu (rys. 2.4). Dopiero zachowanie należytej staranności w procesie konstrukcji ontologii umożliwi ich poprawną integrację.



Rysunek 2.4: Procedury odwzorowywania i łączenia ontologii

Na jakość integracji wiedzy duży wpływ ma cały proces wytwarzania ontologii. Podstawowym warunkiem jest zapewnienie poprawności i spójności ontologii wejściowych. Unikanie typowych błędów [65] czy opieranie się na ustalonych wytycznych [69] pozwala na stworzenie spójnych ontologii, które w przyszłości nie będą wprowadzać sprzeczności, czy niespójności do zintegrowanej wiedzy. Postępowanie zgodnie z różnego rodzaju wskazówkami, opisującymi jak tworzyć czytelne i elastyczne ontologie [65], powoduje, że tworzona ontologia jest kompletna, przez co ma szansę wprowadzić dodatkową wiedzę do integrowanej ontologii.

Konieczne jest również zastosowanie precyzyjnego słownictwa opisującego wiedzę zawartą w ontologiach czy dobór pojęć bazowych, na których oparte będą integrowane ontologie [24].

Trzecim istotnym aspektem jest stosowana w trakcie wytwarzania ontologii metodologia pracy [23, 25]. Zastosowanie jednej z uznawanych metodologii, takich jak Methontology [57], NeOn [107] czy metodologia opracowana przez Noy i McGuiness [112], znacząco wpływa na jakość uzyskanego produktu. Wspomniane metodologie w dużej mierze uwzględniają potrzebę przyszłej

integracji wiedzy, a w połączeniu z narzędziami typu Protégé [62, 110, 132] czy OCS [18, 20, 22], pozwalają na tworzenie spójnych i formalnie oraz logicznie poprawnych ontologii.

Rozdział 3

Przegląd istniejących technik łączenia oraz odwzorowywania ontologii dziedzinowych

3.1 Wprowadzenie

Wiele zespołów podjęło prace na polu integracji ontologii [85, 111]. Wyróżnić można dwa główne podejścia do tego problemu. Jedno z nich zakłada, że procesowi tworzenia ontologii towarzyszy mniej lub bardziej świadoma myśl, iż w przyszłości będą one częścią większej całości. Podejście to zakłada istnienie pewnej metaontologii, która będzie rozszerzana przez wszystkie inne ontologie. Wzajemne odwzorowanie dowolnych dwu ontologii w takiej sytuacji sprowadzałoby się jedynie do jednorazowego stworzenia odwzorowań do metaontologii. Drugie podejście zakłada pełną lub częściową automatyzację procesu dynamicznego tworzenia odwzorowań pomiędzy dwiema ontologiami. Zakłada ono leksykalną i/lub semantyczną analizę obu ontologii w celu znalezienia najlepszego dopasowania klas obu ontologii.

W rozdziale przedstawiono przegląd istniejących technik łączenia i odwzorowywania ontologii. W pierwszej części rozdziału przedstawiono zarówno algorytmy analizy syntaktycznej, jak i semantycznej analizy podobieństwa pomiędzy etykietami pojęć zawartych w ontologiach.

W dalszej części zaprezentowano koncepcję metaontologii, czyli konstrukcji stanowiących swoisty wysokopoziomowy „szkielet" spinający szczegółowe ontologie dziedzinowe.

W trzeciej części przedstawiono istniejące opracowania mające na celu dynamiczną konstrukcję powiązań pomiędzy elementami ontologii poprzez analizę podobieństwa zarówno semantycznego, leksykalnego, jak i strukturalnego.

Opisane w tym rozdziale algorytmy stanowią pozycję wyjściową dla konstrukcji proponowanego algorytmu łączenia i odwzorowywania ontologii dziedzinowych opisanego w kolejnym rozdziale.

3.2 Integracja ontologii

Ontologie w założeniu powinny modelować wybraną domenę w obiektywny sposób. Literatura pokazuje jednak, że konstrukcja ontologii, podobnie jak i inne formy reprezentacji wiedzy, w dużej mierze zależy od przyjętych przez autora założeń czy uproszczeń [28]. System komputerowy, wykorzystujący pewną ontologię, komunikujący się z innym systemem również wykorzystującym

ontologię, może opierać się na ontologii całkowicie odmiennej, nawet pomimo zastosowania tego samego języka opisu świata. Wysoce prawdopodobne jest wręcz, że używa on zupełnie odmiennej terminologii, poziomu dokładności struktury semantycznej, czy liczby i rodzaju atrybutów klas. Takie systemy komputerowe musiałyby być w stanie wywnioskować znaczenie elementów obu ontologii a następnie wyszukać elementy najbardziej sobie odpowiadające. Pożądane jest również by były w stanie scalać elementy integrowanych ontologii lub dokonywać ich faktoryzacji. W przeciwnym wypadku ich współpraca lub integracja rozwiązań o różnym poziomie szczegółowości byłyby znacznie utrudnione.

Zarówno w przypadku łączenia, jak i odwzorowywania ontologii najbardziej podstawowym mechanizmem jest ręczne tworzenie odwzorowań i złączeń ontologii. Niestety proces ten szybko rośnie do rangi problemu niemożliwego do wykonania ze względu na liczbę istniejących rozwiązań wymagających integracji. Ponadto każda zmiana jednej z ontologii wymagałaby modyfikacji wszystkich wcześniej wykonanych odwzorowań tej ontologii w inne. Z kolei automatyzacja tego procesu napotyka na trudności ze względu na różnice w postrzeganiu świata przez różnych twórców.

W poprzednim rozdziale zauważono, że zarówno proces łączenia, jak i odwzorowywania ontologii, charakteryzuje się podobnymi problemami napotykanymi przez badaczy, a etap tworzenia odwzorowań można potraktować jako etap poprzedzający konstrukcji złączenia ontologii. Dla uproszczenia rozważań, w dalszej części tej pracy poprzez łączenie czy integrację ontologii rozumiane będzie zarówno tworzenie odwzorowań pomiędzy jej elementami, jak i tworzenie nowej ontologii będącej wynikiem procesu integracji łączonych ontologii.

Nieodzownie z procesem integracji wiedzy wiąże się zbieżność czy podobieństwo integrowanych konstrukcji. Na problem podobieństwa ontologii spogląda się dwojako – rozważane jest podobieństwo całych ontologii jak i tylko elementów wchodzących w ich skład. W literaturze, ze względów pragmatycznych, przeważa to drugie podejście, gdyż ontologie jako całość mogą zawierać np. odmienną liczbę konceptów, z których tylko część jest ze sobą w jakimś stopniu związana. W celu przeprowadzenia procesu integracji takich ontologii nie jest konieczna analiza podobieństwa między ontologiami a bezpośrednio elementów podlegających porównaniu. Podobieństwo pomiędzy zarówno ontologiami jak i elementami ontologii mierzy się na podstawie dwu aspektów: syntaktycznego i semantycznego. W tym pierwszym przypadku zwracamy uwagę na analizę składniową ontologii czy zawarte w niej relacje pomiędzy pojęciami. W drugim przypadku nacisk kładziony jest na podobieństwo znaczeniowe konceptów, a główną rolę odgrywa zbieżność informacji zawartych w porównywanych pojęciach. Oba te podejścia zostaną szerzej zaprezentowane w dalszych częściach tego rozdziału.

3.3 Opis istniejących algorytmów analizy syntaktycznej podobieństwa ontologii

Popularnym mechanizmem wyznaczania podobieństwa ontologii jest jej analiza syntaktyczna. Zaletą tego podejścia jest możliwość operowania na formalnych i jednoznacznych strukturach w oderwaniu od, często trudnego do określenia, znaczenia stojącego za konceptami. W podejściu tym zauważono, że wiedza zapisana w postaci ontologii jest grafem skierowanym, którego wierzchołkami są pojęcia, a krawędziami relacje zachodzące pomiędzy tymi pojęciami. Stąd wiele operacji służących do porównywania ontologii opiera się na różnych algorytmach operacji na grafach.

Część zespołów swoją pracę opiera na grafach konceptualnych (ang. *conceptual graphs*). Jest to system opisu logiki pierwszego rzędu opracowany przez Johna F. Sowę [129, 130]. Siła grafów konceptualnych polega na możliwości opisu znaczenia w postaci zrozumiałej zarówno dla człowieka jak i maszyny. Mają też one bezpośrednie odwzorowanie do języka naturalnego. Ze względu na tę właściwość stosowane są jako forma pośrednia przy przekształcaniu sformalizowanego zapisu komputerowego na/z języka naturalnego. Grafy konceptualne znalazły szerokie zastosowanie w projektach mających na celu pozyskiwanie informacji oraz przetwarzanie języka naturalnego czy też w systemach eksperckich.

Rysunek 3.1: Graf konceptualny odpowiadający zdaniu „Kot leży na macie".

Na rys. 3.1 przedstawiono graf odpowiadający prostemu zdaniu „Kot leży na macie". Prostokątami oznaczone są koncepty występujące w tym zdaniu („Kot" oraz „Mata"), okręgami bądź owalami relacje jakie pomiędzy tymi konceptami występują („leży"), a strzałkami kierunek tej relacji. Graf ten też można zapisać w postaci liniowej (wyrażenie 3.1).

```
[Kot]->(leży)->[Mata]
```
(3.1)

Zarówno postać liniowa jak i grafowa służą do komunikacji pomiędzy ludźmi lub pomiędzy człowiekiem a maszyną. Do komunikacji opracowano postać zapisu CGIF (ang. *conceptual graph interchange form*, wyrażenie 3.2). Koncepty wiązane są relacjami za pośrednictwem tzw. etykiet współpracujących (ang. *coreference labels*) `x` oraz `y`.

```
[Kot:  *x] [Mata:  *y] (leży ?x ?y)
```
(3.2)

Jedną z prób zastosowania grafów konceptualnych do łączenia ontologii były prace zespołu pod kierownictwem Dienga i Huga [46]. Opracowany algorytm opiera się na dwóch etapach: analizie leksykalnej elementów grafów oraz na analizie relacji, w jakich biorą udział elementy wykryte w pierwszym etapie. Analiza leksykalna opiera się na prostym porównaniu etykiety jednego elementu z etykietą i synonimami drugiego elementu. Na tym etapie algorytm nie analizuje kontekstu w jakim umieszczony jest badany element. Dopiero na drugim etapie, poprzez grafy konceptualne analizowana jest struktura relacji i kontekst, w jakim ulokowane są porównywane elementy.

Dalsze prace nad wykorzystaniem grafów konceptualnych podjęła Madalina Croitoru wraz z zespołem [37]. Opracowano teoretyczny model pozwalający określić podobieństwo ontologii bazując na projekcji grafu konceptualnego jednej ontologii w graf drugiej ontologii. Niestety rozwiązanie to funkcjonuje obecnie jedynie w przestrzeni teoretycznej, gdyż brak jest implementacji potwierdzających empirycznie opracowaną teorię.

O wiele prostsze rozwiązanie zastosowano przy pracach nad algorytmem porównywania ontologii dla systemu BioAgent rozwijanego przez Uniwersytet w Camerino we Włoszech [38]. Zespół z tego uniwersytetu założył, że podobieństwo ontologii wynika z podobieństwa ich elementów. Koncepty porównywane są na podstawie liczby i typu relacji w jakich biorą udział. Celem tego rozwiązania jest umożliwienie agentom komputerowym integracji wiedzy z różnych platform, gdzie tzw. rdzeń ontologii, czyli podstawowy zestaw pojęć, na każdej platformie jest identyczny. Występuje tutaj więc założenie mocno ograniczające zastosowanie opracowanej miary podobieństwa elementów ontologii do sytuacji, gdzie porównywane zbiory wiedzy wywodzą się z jednego, wspólnego przodka.

Niektórzy badacze traktują porównywane ontologie jako zbiory rozważając pokrywanie się kotopii (ang. *cotopy*) pojęć [100], czyli zbiorów zawierających wszystkich rodziców i potomków analizowanych elementów. Podejście to reprezentują w swojej pracy Maedche i Staab [99]. Do porównywania samych pojęć stosują zdefiniowaną przez Levenshtein'a odległość edycyjną (ang. *edit distance*) [94], czyli liczbę przestawień, wstawień oraz usunięć niezbędnych, by przekształcić jeden ciąg znakowy na inny. Zastosowana metoda przetestowana przez autorów w warunkach laboratoryjnych wykazała przydatność podejścia do wyznaczania podobieństwa ontologii i umożliwia odgórne spojrzenie na problem integracji ontologii.

Również Zhao i Halang [145] wyznaczając miarę podobieństwa ontologii traktują je jako zbiory. Łącząc teorię zbiorów przybliżonych [117] oraz teorię przestrzeni konceptualnej [61] opracowali algorytm wyznaczania podobieństwa elementów ontologii za pomocą modelu Tversky'ego [124, 139]. W wyniku czego powstało rozwiązanie rozbudowane matematycznie i bardzo silne z formalnego punktu widzenia.

Analiza syntaktyczna, mimo że oparta jest na formalnych strukturach i dobrze opisanych językach, nie umożliwia pełnego przeprowadzenia procesu integracji wiedzy. Nietrudno sobie wyobrazić sytuację, gdzie 2 ontologie, opisujące całkowicie odmienne dziedziny, charakteryzować będą się identyczną strukturą powiązań pomiędzy konceptami. Bez posiłkowania się analizą semantyczną algorytm wykazałby pełną zgodność integrowanych ontologii prowadząc do błędnego rozwiązania. Z tego powodu większość algorytmów opisanych w tej części pracy do poprawnego działania wymaga wsparcia algorytmów semantycznej analizy zawartej w ontologiach wiedzy.

3.4 Opis istniejących algorytmów analizy semantycznej podobieństwa ontologii

Analiza semantyczna dostarcza wielu nowych możliwości pod względem analizy syntaktycznej. Podejście to umożliwia uwzględnienie sensu przechowywanej w ontologii informacji a nie jedynie jej struktury. Z drugiej jednak strony stoi przed problemem analizy języka naturalnego, którego bogactwo, wielość form wyrazu, stopień komplikacji czy uzależnienie od kontekstu są przeszkodą w automatyzacji tego procesu.

W dużej mierze działanie tych algorytmów uzależnione jest od dostępności rozbudowanych słowników umożliwiających znaczeniowe powiązanie analizowanych konceptów. Dynamiczny rozwój algorytmów umożliwiających semantyczną analizę tekstu, czy określania znaczenia słów, umożliwiło pojawienie się słownika kontekstowego WordNet [54]. Jest on leksykalną bazą danych utworzoną i rozwijaną dla języka angielskiego. W odróżnieniu od tradycyjnych słowników operuje on na znaczeniach słów (tzw. *synset*), a nie na samych słowach. Synset oznacza tutaj grupę słów będących synonimami. Różne znaczenia danego słowa zawsze ujęte są w różnych synsetach. Synsety z kolei są pomiędzy sobą powiązane relacjami, co w wyniku tworzy sieć konceptów powiązanych ze sobą znaczeniowo.

Możliwości oferowane przez WordNet pozwoliły na opracowanie wielu rozwiązań określających podobieństwo elementów ontologii. Można je podzielić na 3 główne kategorie:

1. Zliczające krawędzie pomiędzy konceptami – algorytmy te określają podobieństwo na podstawie liczby krawędzi w grafie powiązań słownika WordNet łączących analizowane koncepty. Bazują na spostrzeżeniu, że im bardziej spokrewnione pojęcia, tym więcej połączeń występuje pomiędzy nimi. Algorytmy te osiągają różną skuteczność [140] i bazują na różnych miarach. Najprostsze algorytmy obliczają najkrótszą ścieżkę [32, 123], inne podobieństwo uzależniają od położenia pojęć względem najbliższego wspólnego rodzica (np. Wu and Palmer [144]) czy wprowadzają nieliniową funkcję wiążącą odległość pomiędzy pojęciami z ich odległością od najbliższego wspólnego rodzica (np. Li et al. [95]). Testy wykazują bardzo wysoką dokładność i prędkość działania ostatniego rozwiązania.

2. Bazujące na zawartości informacyjnej – klasa algorytmów określających podobieństwo pomiędzy pojęciami bazując na ilości współdzielonej pomiędzy nimi informacji. Każdemu z pojęć w taksonomii przypisuje się prawdopodobieństwo bycia powiązanym z analizowanym konceptem. Najprostsze algorytmy tego typu zwracają po prostu wartość prawdopodobieństwa najbliższego wspólnego rodzica (Lord et al. [98]). Miara podobieństwa opracowana przez Resnik'a [122] określa z kolei zawartość informacyjną najbliższego wspólnego rodzica. Wadą tego rozwiązania jest fakt zwracania wartości z nieokreślonego przedziału – im większe podobieństwo tym wyższa zwrócona wartość, jednak nie ma z góry ustalonego maksimum. Z podobnym problemem boryka się algorytm opracowany przez Jiang i Conrath'a [82]. Autorzy zaproponowali wzbogacenie analizy zawartości informacyjnej wiedzą wynikająca z liczby i struktury połączeń pomiędzy konceptami. Podejście to dało bardzo dobre rezultaty, jednak brak ograniczenia co do zakresu zwracanych wartości ogranicza jego zastosowanie. Analiza opracowań wskazuje, że najlepszym algorytmem tej klasy jest rozwiązanie opracowane przez Lin [96, 97]. Autor swoją miarę podobieństwa opiera na złączeniu informacji niezbędnej

do pełnego opisania porównywanych pojęć, jak i informacji niezbędnej do pełnego opisania części wspólnej kryjących się za nimi znaczeń.

3. Bazujące na disambiguacji znaczenia pojęć – algorytmy te bazują na algorytmie Lesk'a [93]. Algorytm ten wykorzystuje kontekst, w jakim zostało użyte słowo do określenia jego sensu, głównie na podstawie zliczania liczby słów współdzielonych przez definicje wyrazów znajdujących się w analizowanym kontekście. Banerjee i Pedersen dopracowali i rozszerzyli ten algorytm opracowując miarę opartą na analizie znaczeń porównywanych konceptów [10].

Każda z powyższych klas algorytmów posiada swoje mocne i słabe strony. W zależności od danych wejściowych osiągają różną jakość wartości podobieństwa pojęć. Badacze rozważają również algorytmy oparte na kombinacji metod z różnych kategorii (np. Abu Helou i Abid [1]) po części eliminując problem różnorodności danych wejściowych. Najlepsze algorytmy z każdej kategorii osiągają korelację do testu Millera i Charlesa [104] bliską granicznej, wynoszącej 0,9, zdefiniowanej przez Resnik'a [122], a będącej wynikiem pomiarów porównań wykonanych przez ludzi. Algorytm opracowany przez Li osiąga korelację 0,82 [122], a zaproponowane przez Lin oraz Jiang i Conrath'a 0,83 [97] [82]. Najsłabiej wypada trzecia kategoria algorytmów opartych na disambiguacji słów, korelacja algorytmu Banerjee i Pedersena wynosi 0,75. Również czas wykonania tego algorytmu jest najdłuższy [1]. Ponadto wszystkie trzy powyższe kategorie algorytmów, poprzez korzystanie z zewnętrznego źródła informacji w trakcie procesu wyznaczania podobieństwa elementów, są niezależne od języka, sposobu reprezentacji czy poziomu logiki użytego w integrowanych ontologiach.

Odmienne podejście zastosowali Rudi Araújo oraz Sofia Pinto [5, 6]. Podobieństwo semantyczne jest tu rozumiane w aspekcie logiki formalnej, a nie pragmatycznym, jak to miało miejsce w algorytmach przedstawionych wcześniej w tej sekcji. Miara podobieństwa wyznaczana jest na podstawie zgodności pojęć charakterystycznych wchodzących w skład obu ontologii, gdzie poprzez charakterystykę pojęć oznaczamy najbardziej szczegółowe pojęcie wywodzące się z porównywanych konceptów. Rozwiązanie przedstawione przez Araújo oraz Pinto oparte zostało na wnioskerze #SAT [12, 138], poprzez redukcję subsumpcji pojęć charakterystycznych do koniunkcyjnej postaci normalnej (ang. *CNF - conjunctional normal form.* Wniosker ten działa w oparciu na problemie k-spełnialności binarnej, będącym problemem NP-trudnym. Ogranicza to rozmiary ontologii, dla których rozwiązanie może zostać zastosowane. Ponadto algorytm może zostać zastosowany jedynie do ontologii wyrażonych w formalnych językach logiki opisowej.

Zespół pod kierownictwem Alexa Borgidy próbował połączyć przedstawione powyżej podejścia [27]. W swojej pracy zaproponował uogólnienie istniejących algorytmów operujących na konceptach prostych, rozumianych pragmatycznie, tak by mogły operować na konceptach złożonych, będących podstawą logik opisowych. Zespół wykazał, że możliwe jest takie przekształcenie dla wielu miar podobieństwa elementów ontologii. Wiele logik, zwłaszcza bardziej złożonych, wymaga jednak do poprawnego działania probabilistycznych odpowiedników słownika WordNet. Jako przykład takiego słownika podano ontologię P-Classic [91], jednak, jak sami autorzy zauważają, budowa kompletnej ontologii tego typu jest mało prawdopodobna. Rozwiązaniem problemu może tutaj być rozszerzenie słownika WordNet o dane probabilistyczne, lecz, zdaniem Borgida, wymagało by to dokonania dość znacznych uproszczeń, a same wartości przypisane konceptom miałyby małą dokładność.

Problem analizy semantycznego podobieństwa ontologii i ich elementów jest dobrze rozpoznany przez badaczy. Oferowane rozwiązania skupiają się zarówno na podejściu pragmatycznym, jak i formalnym, a także łączą oba te punkty widzenia. W każdym przypadku działanie algorytmów uzależnione jest od jakości zewnętrznego słownika wspierającego proces wyznaczania podobieństwa konceptów. Dzięki dynamicznemu rozwojowi słownika WordNet oraz jego pochodnych (dla innych języków naturalnych niż angielski) pragmatyczne i nieformalne podejście do procesu wyznaczania podobieństwa konceptów podlega dynamicznemu rozwojowi. Formalne rozwiązania, bazujące na logice opisowej, posiadają znacznie bardziej rozbudowany zapis teoretyczny, lecz ograniczone są brakiem narzędzi wspomagających, czy koniecznością dostosowania do konkretnej logiki opisowej użytej w konstrukcji łączonych ontologii.

3.5 Opis istniejących algorytmów łączenia oraz odwzorowywania ontologii podobnych

3.5.1 Metaontologie

Metaontologie, czyli ustandaryzowane, formalne ontologie, składające się z abstrakcyjnych pojęć lub obejmujące swoim zakresem szeroki zbiór konceptów, są więc postrzegane przez część badaczy jako rozwiązanie dla problemu heterogeniczności wiedzy. Zazwyczaj są to konstrukcje interdyscyplinarne, umożliwiające integrację wiedzy z różnych dziedzin, czy też współistnienie różnych ontologii o tej samej dziedzinie, lecz prezentujące odmienne spojrzenie na wybrane zagadnienie. Już samo zastosowanie wspólnej bazy podstawowych pojęć przy konstrukcji ontologii opartych o różne źródła wiedzy zapewnia znaczne podobieństwo ontologii [24]. Rozszerzenie tego zbioru o elementy formalne jest kuszącym podejściem skłaniającym coraz więcej badaczy do zajmowania się tym problemem. Metaontologie są swoistym spoiwem dla wszystkich ontologii utworzonych na ich podstawie.

Formalne metaontologie

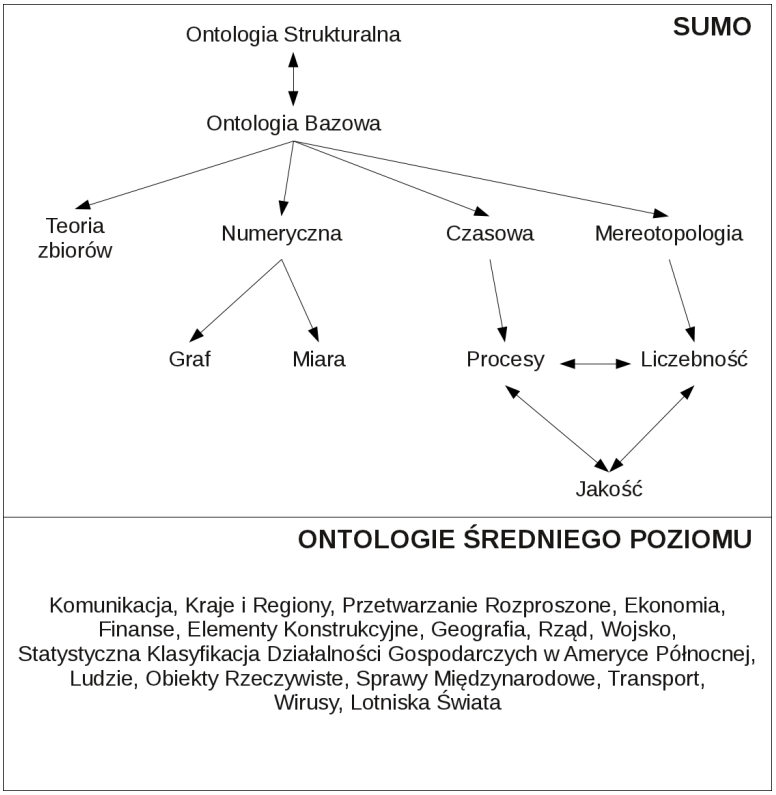
Jedną z najpopularniejszych metaontologii jest The Suggested Upper Merged Ontology (SUMO) [109]. Prace nad tą ontologią oryginalnie rozpoczęto w ramach organizacji Teknowledge Corporation. SUMO dostarcza podstaw dla tworzenia ontologi dziedzinowych i średniego poziomu. Celem tego projektu jest promocja interoperacyjności danych, pozyskiwania informacji czy mechanizmów przetwarzania języka naturalnego. Obecnie ontologia SUMO [77] składa się z 4000 aksjomatów i 1000 pojęć. SUMO obejmuje m.in. następujące obszary:

- pojęcia strukturalne, takie jak klasa, podklasa, byt,
- ogólne typy obiektów i procesów,
- abstrakcje takie jak teoria zbiorów, atrybuty, relacje,
- liczby i miary,
- pojęcia definiujące czas, np. czas trwania,
- części i całości,
- relacje semiotyczne,
- intencjonalność i inne.

Wysokopoziomową strukturę ontologii SUMO i ontologii na niej opartych przedstawia rys. 3.2.

Relatywnie niewielka liczba konceptów ma w założeniu uprościć zrozumienie i zastosowanie utworzonych pojęć i aksjomatów.

Odmienne podejście zastosowała firma Cycorp [41], odpowiedzialna za ontologię Cyc oraz OpenCyc, będącą otwartym, publicznie dostępnym odpowiednikiem ontologii Cyc. Projekt Cyc ma na celu wytworzenie prawdziwej sztucznej inteligencji i został utworzony przez Douglasa Lenata już w 1984 roku. Ontologie są na uwadze organizacji od 1994 roku. Od tego czasu udało się utworzyć ogromną ontologię składająca się z setek tysięcy konceptów powiązanymi milionami aksjomatów. W ten sposób powstała metaontologia, której dziedziną jest cała otaczająca ludzi rzeczywistość. W wersji 2.0, oprócz relacji dziedziczenia oraz instancji wprowadzono również pojęcie „szerszego zakresu" (ang. *broader term*) pozwalające na lepsze zdefiniowanie zależności pomiędzy konceptami, a także dodano powiązania pojęć z odpowiadającymi im artykułami w popularnej encyklopedii jaką jest Wikipedia [143]. Praktycznie od samego początku prac autorom ontologii Cyc przyświeca idea wyrażona przez Lenata w 2001 roku: „Kiedy zbierzesz odpowiednio dużo informacji i zintegrujesz



Rysunek 3.2: Wysokopoziomowa struktura ontologii SUMO i związanych z nią ontologii pochodnych [118]

ją jako wiedzę, wtedy oprogramowanie osiągnie status nadczłowieka w takim sensie, jak ludzkość posiadająca zdolność pisania składa się z nadludzi w stosunku do ludzkości pozbawionej tej umiejętności" [39]. W swojej obecnej postaci OpenCyc może z powodzeniem zostać zastosowany w różnego rodzaju aplikacjach inteligentnych, takich jak:

1. rozumienie mowy,
2. integracja baz wiedzy,
3. sprawdzanie spójności danych,
4. rozwój ontologi poprzez rozbudowę bazy danych zawartą w OpenCyc,
5. sterowanie ruchem sieciowym i inne.

Na potrzeby projektu UMBEL (ang. *Upper Mapping and Binding Exchange Layer*) [17], mającego na celu utworzenie odwzorowania zasobów internetowych do jednej standardowej ontologii, oszacowano przydatność ontologi Cyc do realizacji tego typu zadań. Przygotowany przez Marka Bergmana raport [16] opisuje zalety jak i wady produktu firmy Cycorp. Raport ten wymienia następujące zalety ontologii OpenCyc:

1. Stabilność – prace nad ontologią pochłonęły około 200 osobolat, a sama uzyskana struktura była wielokrotnie testowana i weryfikowana poprzez liczne aplikacje. Ponadto rozwój bazy wiedzy Cyca wspierany jest poprzez internetową grę FACTory [40], umożliwiającą wprowadzanie nowych informacji każdemu internaucie,

2. Społeczność – Cyc ma szerokie grono użytkowników wywodzące się z organizacji akademickich, rządowych, komercyjnych, czy non-profit. Udział w projekcie możliwy jest również dla wolontariuszy poprzez The Cyc Foundation,

3. Aktualizacje – opracowano mechanizmy aktualizacji OpenCyc do szerszego ResearchCyc, pełnej ontologii Cyc czy innych usług dostarczanych przez Cycorp,

4. Kompleksowość – OpenCyc jest najbardziej rozbudowany spośród obecnie istniejących systemów pod względem zakresu i reprezentatywności pojęć znanych człowiekowi,

5. Konsensus – od momentu swojego powstania projekt Cyc miał na celu wyłuskanie wspólnego konsensusu wynikającego z myślenia ludzkości. Podejście to, z jednej strony bardziej pragmatyczne, niesie za sobą wiele problemów. Starano się skodyfikować logikę nieformalną stojącą za ludzkim pojmowaniem rzeczywistości. Proces ten można porównać do uczenia dziecka chodzenia, czy czytania – jest to rzecz długotrwała i odbywa się na zasadzie prób i błędów. Jednak po uzyskaniu pewnych efektów uzyskuje się trwałą podstawę niezbędną do dalszego rozwoju,

6. Siła wyrazu – celem bazy wiedzy jest umożliwienie komputerom wnioskowania nawet w przypadku znajomości niewielkiego zbioru faktów czy zależności. W skład ontologii Cyc wchodzą tysiące mikroteorii dając w wyniku siłę i pokrycie dziedzinowe unikatowe wśród innych systemów,

7. Otwarte i bezpłatne rozwiązanie – w 2002 roku ontologia Cyc została wydana na zasadach Open Source jako OpenCyc. Ponadto ośrodki badawcze mogą korzystać z rozszerzonej wersji zwanej ResearchCyc. Otwartość rozwiązania jest istotna z punktu widzenia akceptacji rozwiązania, wiele instytucji badawczych unika komercyjnych rozwiązań.

Raport też wskazuje na wady tego rozwiązania:

1. Niejasna metaontologia – z powodu długotrwałego rozwoju ontologia Cyc nie jest tak uporządkowana jak inne podobne rozwiązania. Na szczególną uwagę zasługują elementy potomne pojęcia „Byt" (ang. *Thing*) czy poziomy „Namacalności" (pojęcia definiujące „namacalność" (ang. *tangibility*) elementu na wielu poziomach,

2. Nadmiarowość – dwadzieścia lat prac na różnymi domenami (wiele z nich dla organizacji wojskowych i wywiadowczych) spowodowało powstanie wielu konceptów zbędnych z punktu widzenia cywilnego odbiorcy. UMBEL szacuje, że ich ilość jest dość znaczna i sięga 30% wszystkich pojęć znajdujących się w OpenCyc. Kolejne 15% pojęć to pojęcia abstrakcyjne niezbędne do wnioskowania na podstawie kolorów, wyznania, rozmiaru obiektów, ich położenia w przestrzeni itp.

3. Zbyt duża siła wyrazu – oparta na języku LISP ontologia Cyc złożona jest z wielu konstrukcji logiki wyższego rzędu. Paradoksalnie może to utrudniać konwersję do rozwiązań wykorzystujących prostsze konstrukcje,

4. Przestarzałe konwencje – długi czas rozwoju powoduje, że część założeń nie przystaje do obecnych standardów. Ontologia Cyc nie została np. oparta na modelu trójek znanych z RDF a będących podstawą większości obecnych języków zapisu ontologii. Dostępne jest tłumaczenie ontologii na język OWL, ale ze względu na mniejszą siłę wyrazu języków RDF i OWL tłumaczenie to posiada błędy i wymaga dopracowania,

5. Dokumentacja – brak centralnego repozytorium przechowującego kompletną dokumentację, a materiały nie są przechowywane w spójnej formie. Brakuje również samouczków (ang. *tutorial*) umożliwiających zapoznanie się z samą ontologią i przyjętymi w niej założeniami,

6. Byty nazwane (ang. *named entities*) – tworzona ontologia główny nacisk kładzie na kompletność strukturalną, brakuje w niej bytów nazwanych, a te zawarte dobierane są nierównomiernie do różnych dziedzin.

Na podstawie powyższych obserwacji można stwierdzić, że Cyc jest bardzo dojrzałym rozwiązaniem. Pomimo długiego rozwoju możliwa jest integracja z obecnymi standardami, a największym problemem jest nadmiarowość, która utrudnia zrozumienie wewnętrznej struktury ontologii stanowiącej o sile Cyca. Z tych powodów, pomimo pewnych słabości, ontologia OpenCyc wraz z SUMO posłużyły jako dokumenty wyjściowe dla IEEE Standard Upper Ontology Working Group [76, 108]. Organizacja ta ma na celu opracowanie szeroko akceptowanego standardu, na podstawie którego możliwe będzie tworzenie dowolnych ontologii. Ich współpraca możliwa ma być właśnie za pośrednictwem wspólnej metaontologii, do której każde rozwiązanie nawiązuje w swojej konstrukcji.

Pomimo dużego zaawansowania, prace nad standardową metaontologią są na bardzo wczesnym etapie. IEEE Standard Upper Ontology Working Group obecnie opracowuje dokumenty będące założeniami do dalszych prac. Nie jest również pewne czy w ogóle możliwe będzie utworzenie wspólnej ontologii, która byłaby w stanie zapewnić odpowiedni poziom szczegółowości nawet w obrębie jednej dziedziny problemowej. Elementy nie ujęte w metaontologii będą musiały być integrowane osobno lub dodane do metaontologii. Problemem pozostają również wszystkie ontologie już będące w użytku, a nie wywodzące się z którejkolwiek z zaproponowanych metaontologii. Takich ontologii jest bardzo dużo a codziennie powstają nowe. Stąd zastosowanie metaontologii w codziennych zastosowaniach jest ograniczone i stosowane głównie przy dużych projektach.

WordNet jako metaontologia

WordNet jest leksykalną bazą danych utworzoną i rozwijaną oryginalnie dla języka angielskiego. WordNet wyróżnia rzeczowniki, czasowniki, przymiotniki i przysłówki. Pojęcia powiązane są w synsety, a pomiędzy synsetami występują relacje określające ich wzajemne zależności, co w wyniku tworzy sieć koncepcji powiązanych ze sobą znaczeniowo.

W WordNet ujęto następujące relacje:

- rzeczowniki:
  - nadrzędność (hypernim) – Y jest hypernimem X, jeżeli każdy X jest typu Y,
  - podrzędność (hyponim) – Y jest hyponimem X, jeżeli każdy Y jest typu X,
  - typy skoordynowane (ang. *coordinate terms*) - Y jest skoordynowany z X, jeżeli X i Y mają wspólny hypernim,
  - holonim – Y jest holonimem X, jeżeli X jest częścią Y,
  - meronim – Y jest meronimem X, jeżeli Y jest częścią X,
- czasowniki:
  - nadrzędność (hypernim) – Y jest hypernimem X, jeżeli czynność X jest typu Y,
  - troponim – Y troponimem X, jeżeli czynność Y jest szczególnym przypadkiem czynności X,
  - zawieranie – czasownik Y zawiera się w X, jeżeli wykonując czynność X musimy wykonać czynność Y,
  - typy skoordynowane (ang. *coordinate terms*) – czasowniki współdzielące hypernim,
- przymiotniki:
  - relacyjność,
  - podobieństwo,

  - imiesłowy,
- przysłówki:
  - wspólny rdzeń przymiotnikowy.

Dzięki takiej strukturze WordNet zaczął być postrzegany jako ontologia. W odróżnieniu od innych rozwiązań, takich jak Cyc czy SUMO relacje w nim zawarte nie zostały sformalizowane, co ogranicza ich precyzyjność. Pomimo tego, w oparciu o WordNet powstało wiele rozwiązań przydatnych w procesie analizy języka naturalnego, czy integracji formalnych ontologii. Na szczególną uwagę zasługują miary podobieństwa semantycznego elementów ontologii opisane w poprzednim rozdziale. Mimo, że nieformalna, to struktura powiązań pomiędzy słowami zdefiniowana w słowniku WordNet umożliwia wiązanie ze sobą pojęć pomimo zastosowania odmiennej bazy leksykalnej. Ponadto, dzięki dostępności szerokiej gamy narzędzi pomocniczych czy miar podobieństwa, nie jest konieczne tworzenie odwzorowań ontologii do WordNet, czy ich konstruowanie z myślą o współpracy z WordNet. Możliwość integracji uzależniona jest od dokładności i bogactwa samego słownika WordNet, który poprzez ciągły rozwój umożliwia integrację coraz większej liczby ontologii.

3.5.2 Dynamiczne tworzenie powiązań pomiędzy ontologiami

Rozwiązaniem alternatywnym wobec idei metaontologii jest możliwość dynamicznego generowania odwzorowań między dwiema ontologiami, czyli tworzenie powiązań między klasami, bytami oraz relacjami różnych ontologii. W wyniku tej operacji powstaje transformata umożliwiająca przekształcenie jednej ontologii w drugą lub nowa ontologia będąca złączeniem ontologii wejściowych. Tą ścieżką podąża wiele zespołów badawczych.

Hovy [75] zaproponował zbiór algorytmów heurystycznych opierających się głównie na wyszukiwaniu podobieństw lingwistycznych miedzy konceptami oraz analizie leksykalnej opisów konceptów w języku naturalnym. Algorytm mapowania przebiega następująco:

1. Rozdzielenie składowych nazw elementów ontologii,
2. Porównywanie składowych nazw elementów ontologii,
3. Porównywanie liczby i podobieństwa słów wspólnych w opisach elementów ontologii w języku naturalnym.

Wynikiem działania tego algorytmu są sugerowane powiązania pomiędzy ontologiami, które następnie muszą zostać skorygowane i zatwierdzone przez człowieka.

Rozwiązaniem analizującym również strukturę ontologii zapisanych w języku OWL jest to zaprezentowane przez Euzenat i Volchev [53]. W tej propozycji miara rozwiązania jest średnią ważoną pomiędzy podobieństwami takich elementów definicji konceptu, jak typ, czy etykieta konceptu, dziedzina, zakres oraz ograniczenia właściwości, taksonomia itp.

Innym możliwym podejściem jest to reprezentowane przez GLUE [47]. System ten do poszukiwania odwzorowań stosuje techniki uczenia się maszyn. Rożne algorytmy uczące analizują informacje zawarte w instancjach konceptów oraz w taksonomii ontologii. Wyniki te są następnie scalane za pomocą technik probabilistycznych. System ten najlepiej sprawuje się dla ontologii posiadających dużą liczbę instancji oraz gdy atrybuty wypełnione są tekstowymi etykietami, a nie odnośnikami do innych instancji.

Połączenie wielu reguł podobieństwa zaproponowali Ehrig i Sure [48]. Tezą ich pracy jest stwierdzenie, iż zestawienie wielu, wprowadzonych ręcznie przez człowieka, reguł podobieństwa konceptów da lepsze odwzorowanie jednej ontologii w drugą, niż zastosowanie prostych warunków. Rozwiązanie to zapewnia praktycznie automatyczne odwzorowanie oraz łączenie ontologii na podstawie jednorazowo wprowadzonego przez ekspertów zestawu reguł.

Goczyła i Grabowska [66] zaproponowali wydajny algorytm integracji ontologii zapisanych metodą kartograficznej reprezentacji wiedzy (ang. *knowledge cartography*) [67, 68]. Metoda ta zakłada podział ontologii na regiony będące fragmentami nie mającymi elementów wspólnych w obrębie jednej ontologii. W trakcie integracji wiedzy rozważa się jakie jest prawdopodobieństwo występowania elementu w określonym regionie jednej i określonym regionie drugiej integrowanej ontologii. Proponowany algorytm zakłada jednak określoność stosowanej terminologii. W przeciwnym wypadku komputer nie jest w stanie automatycznie wyznaczyć „regionów niespełnialnych" w utworzonej ontologii. Alternatywą jest rozszerzenie ontologii o tzw. „współczynnik spełnialności" oznaczający wartość prawdopodobieństwa występowania osobników w wskazanym regionie. Wymagany jest też zapis ontologii w określonej postaci, jaką jest postać kartograficzna.

Noy i Musen opracowali algorytm i narzędzie o wspólnej nazwie PROMPT [113]. Opracowany algorytm jest mechanizmem półautomatycznym – prowadzi użytkownika poprzez proces integracji wiedzy. W pierwszej kolejności generowane są sugestie powiązań bazujące na nazwach klas. Na podstawie akcji użytkownika, akceptującego lub modyfikującego sugerowane powiązania, algorytm generuje kolejne podpowiedzi uwzględniając już zgromadzoną wiedzę wynikającą z działań użytkownika. Działanie algorytmu porównano z pracą ekspertów. Badania wykazują dużą skuteczność metody – wg badań przeprowadzonych przez autorów eksperci w 90% przypadków podążali za sugestiami algorytmu, a w 75% przypadków zaakceptowali sugerowane przez PROMPT rozwiązania sytuacji konfliktowych. Łącznie eksperci w 74% podążali za działaniami algorytmu, a jedynie w pozostałych 26% nie zgadzali się z nim. Niestety pomimo swojej dużej skuteczności oraz niezależności od formalizmów, na jakich oparte są integrowane ontologie, algorytm do poprawnego działania wymaga udziału człowieka, gdyż dopiero na podstawie jego akcji może podjąć kolejne decyzje co do kształtu integrowanej ontologii.

Innym rozwiązaniem półautomatycznym wspierającym prace człowieka jest Chimaera [101]. Na podstawie analizy leksykalnej etykiet, synonimów czy akronimów konceptów narzędzie to pozwala na zasugerowanie podobnych do siebie elementów upraszczając proces integracji wiedzy. Narzędzie to, podobnie jak PROMPT, niestety nie umożliwia automatycznego procesu łączenia ontologii.

Ze względu na rosnącą liczbę powstających ontologii badania na polu ich łączenia i odwzorowywania są aktywnie prowadzone przez wiele zespołów. Powyżej przedstawiono jedynie kilka reprezentatywnych podejść. Pomimo dużej ich różnorodności, istniejące obecnie rozwiązania są niewystarczające. Wiele algorytmów rozważa podobieństwo leksykalne, a pomija semantyczną zbieżność konceptów. Brakuje również algorytmów prostych i wydajnych na tyle, by mogły być z powodzeniem zastosowane przez różnego rodzaju systemy komputerowe dla dowolnej pary ontologii. Część rozwiązań przyjmuje, że to człowiekowi pozostaje ostateczna decyzja o kształcie odwzorowania, co przeczy idei automatyzacji procesu. Inne skupiają się na formalnym zapisie ontologii pomijając w ten sposób bogactwo form wyrażania tej samej idei dostępnych w języku naturalnym. Część opracowanych rozwiązań obejmuje jednak leksykalny aspekt łączonych ontologii. Stanowią one doskonałą podstawę do dalszych badań. Niestety złożoność i różnorodność języka naturalnego, stosowanego do opisu poszczególnych elementów ontologii, pozostaje nadal nierozwiązanym problemem, zwłaszcza gdy przystępujemy do analizy rozwiązań utworzonych w dwu lub więcej językach. W przypadku ontologii monolingwistycznych rozwiązania takie jak WordNet pozwalają na sprawne określanie relacji między zastosowanymi opisami konceptów.

Rozdział 4

Warunki oraz sposoby łączenia i odwzorowywania ontologii

4.1 Wprowadzenie

W tym rozdziale zidentyfikowano problemy napotykane w trakcie integracji ontologii oraz zaproponowano ich rozwiązania w postaci nowego algorytmu łączenia i odwzorowywania ontologii.

W pierwszej części rozdziału opisano zagadnienie współpracy systemów komputerowych i wynikające z niej problemy przy konstrukcji zarówno odwzorowań, jak i złączeń ontologii opisujących te systemy. W drugiej części dokonano wyboru algorytmów pomocniczych służących do określania podobieństwa pomiędzy konceptami, a w części trzeciej zdefiniowano poziomy tego podobieństwa.

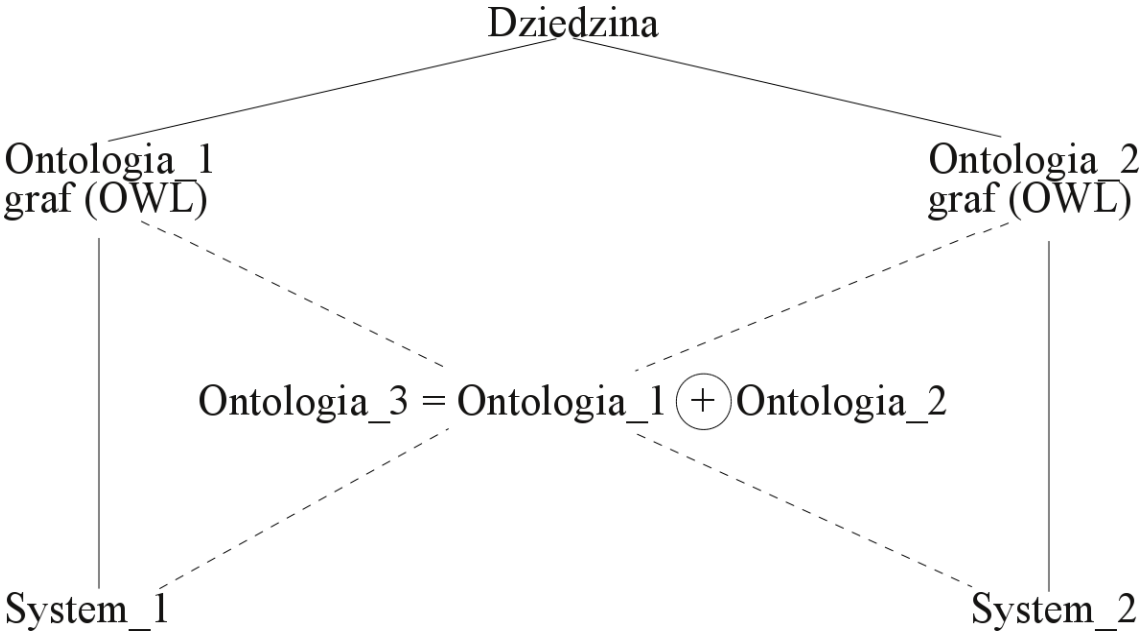
W dalszej części przedstawiono proponowany algorytm oraz zaprezentowano jego działanie na przykładzie integracji ontologii opartych na słownikach ENISA oraz NIST.

W kolejnym rozdziale przedstawiono implementację oraz działanie algorytmu w systemie OCS.

4.2 Problemy napotkane w trakcie integracji ontologii

Łączenie ontologii, czyli tworzenie powiązań między klasami, bytami oraz relacjami rożnych ontologii, zwracające w wyniku nową ontologię, zawierającą wszystkie elementy łączonych rozwiązań, ma na celu zapewnienie interoperacyjności systemów opartych na ontologiach. Obecnie dowolność reprezentacji świata rzeczywistego w systemach komputerowych powoduje, że systemy różnych dostawców napotykają problemy w wymianie informacji. Proces ręcznego tworzenia odwzorowań szybko rośnie do rangi problemu niemożliwego do wykonania ze względu na ilość istniejących rozwiązań oraz ich zmienność. Z kolei automatyzacja tego procesu napotyka na trudności ze względu na różnice w postrzeganiu świata przez rożnych twórców.

System komputerowy oparty na ontologiach nie może sprawnie współpracować z innymi systemami bez świadomości konstrukcji ontologii tych systemów. Integracja ontologii opisujących różne systemy umożliwi ich interoperacyjność bez konieczności (lub tylko przy nieznacznej) modyfikacji ich konstrukcji - za pomocą pojęć znanych jednemu z systemów możliwe będzie odniesienie się do pojęć znanych drugiemu systemowi. Sytuację to prezentuje rys. 4.1. System_1 operuje na podstawie pewnej Ontologii_1, a System_2 pewnej Ontologii_2. Obie te ontologie opisują wspólną dziedzinę, ale w ogólności prezentują na nią odmienne spojrzenie. Dzięki integracji wiedzy zawartej w obu ontologiach następuje uwspólnienie bazy wiedzy, jednak przy zachowaniu zbioru pojęć obu ontologii. Oba systemy mogą więc działać w oparciu na wspólnej Ontologii_3, a tym samym komunikować się ze sobą czy realizować jakieś wspólne zadanie.



Rysunek 4.1: Interoperacyjność systemów z punktu widzenia wykorzystanych ontologii

Zapewnienie interoperacyjności wymaga procesów integracji wiedzy, jaką dysponują oba systemy. Ta z kolei napotyka na szereg problemów (Tablica 4.1). Najpoważniejszym z nich jest bogactwo języka naturalnego i wynikająca z niej mnogość form wyrazu i możliwości zapisania tej samej informacji, czy też zapisania różnych pojęć za pomocą tego samego zwrotu. Aby im sprostać, ograniczono jego działanie jedynie do najbardziej istotnych elementów języka naturalnego jakim są rzeczowniki. Nazwy klas i bytów zazwyczaj są rzeczownikami lub złożeniem słów będącym rzeczownikiem. W związku z tym, w celu ograniczenia złożoności obliczeniowej i logicznej problemu, analiza znaczenia etykiet konceptów ograniczona została do tej części mowy. Problem niejednoznaczności języka algorytm rozwiązuje poprzez założenie dziedzinowości ontologii, a semantyczny słownik WordNet, dzięki rozbudowanym strukturom i relacjom pomiędzy słowami, umożliwia porównywanie znaczenia słów oraz określanie ich wzajemnej zależności semantycznej.

Struktura powiązań pomiędzy słowami zdefiniowana w słowniku WordNet umożliwia wiązanie ze sobą pojęć pomimo zastosowania odmiennej bazy leksykalnej. Zdolność taka jest niezastąpiona przy porównywaniu konceptów opisujących różne spojrzenie na ten sam element świata rzeczywistego, zwłaszcza w połączeniu z obserwacją Zhao i Halanga o konieczności istnienia pewnego podobieństwa między łączonymi ontologiami. Zaobserwowano jednak, że w trakcie procesu łączenia i odwzorowywania ontologii istotne jest nie tyle podobieństwo samych ontologii co ich elementów [24]. Można sobie wyobrazić dwie ontologie o zupełnie różnym poziomie szczegółowości lub pokrywające się tematycznie tylko w pewnym stopniu. Wyszukując zbiory elementów podobnych można przeprowadzić proces ich integracji.

Zaobserwowano ponadto, że dla wielu ontologii pojęcia reprezentujące koncepty wyraża się w postaci rzeczowników, a większość informacji zawartych w ontologii wyrażona jest za pomocą klas i relacji dziedziczenia zachodzących pomiędzy nimi [19]. Dziedzinowość z kolei zwiększa prawdopodobieństwo występowania wspólnej bazy koncepcyjnej służącej do opisu danego wycinka świata [24]. Dzięki temu proponowane podejście, przede wszystkim na podstawie związków między słowami zdefiniowanymi w słowniku WordNet, ale także dzięki leksykalnej i strukturalnej analizie łączonych ontologii, umożliwia dynamiczne tworzenie zintegrowanej ontologii łączącej wiedzę zawartą w ontologiach wejściowych.

Tablica 4.1: Kluczowe aspekty integracji wiedzy

Lp.	Aspekt algorytmu	Proponowane rozwiązanie
1	Mnogość form wyrazu dostępnych w języku OWL	Działania algorytmu ograniczone zostaną tylko do części elementów dostępnych w języku OWL dostarczających najwięcej informacji, jakimi są: klasy, byty, komentarze oraz relacje dziedziczenia, przynależności, tożsamości, rozłączności, unii i przecięcia
2	Brak wspólnej bazy pojęciowej łączonych ontologii	Wykrywanie znaczenia słów jest procesem złożonym, o czym świadczy niska skuteczność istniejących obecnie algorytmów. Zmodyfikowany algorytm Lesk'a, przystosowany do wykorzystania bazy WordNet osiąga skuteczność około 32% [9, 11]. W związku z tym przyjęto założenie, że łączone ontologie opisują tę samą dziedzinę
3	Bogactwo form językowych dostępnych w języku naturalnym	Nazwy klas i bytów zazwyczaj są rzeczownikami lub złożeniem słów będącym rzeczownikiem. W związku z tym, w celu ograniczenia złożoności obliczeniowej i logicznej problemu, analiza ograniczona zostanie do tej części mowy
4	Niejednoznaczność języka naturalnego	Semantyczny słownik WordNet dzięki rozbudowanym strukturom i relacjom pomiędzy słowami umożliwia porównywanie znaczenia słów oraz określanie ich wzajemnej zależności semantycznej

4.3 Dobór algorytmów pomocniczych

Sam słownik semantyczny, jakim jest WordNet, oraz skomplikowana struktura powiązań pomiędzy zawartymi w nim pojęciami, nie są wystarczające do rozwiązania problemu integracji wiedzy. Algorytm łączący lub odwzorowujący ontologie musi być w stanie określić podobieństwo między dwoma dowolnymi pojęciami (zarówno semantycznie, jak i niezależnie od słownika WordNet) oraz określić podobieństwo zbiorów pojęć (np. zdań będących komentarzami, czy też pojęć opisanych wieloma wyrazami).

4.3.1 Podobieństwo pomiędzy pojęciami

Do realizacji tego zadania wybrano algorytm opracowany przez Dekang Lin [96, 97]. Zaletą tego algorytmu jest wspomniana w sekcji 3.4 wysoka korelacja do testu Millera i Charlesa. Sam algorytm określa podobieństwo w ustalonym przedziale $< 0, 1 >$, upraszczając tym samym analizę porównawczą wyników porównań różnych par pojęć.

W odróżnieniu od innych algorytmów określających podobieństwo semantyczne, algorytm opracowany przez Lin opiera się nie na empirycznych doświadczeniach, a na zbiorze założeń wynikających z następujących przesłanek:

- podobieństwo pomiędzy pojęciami A i B związane jest z ich wspólnością znaczeniową (ang. *commonality*), czyli ilością wspólnej informacji, jaką niosą za sobą pojęcia A i B. Im ona większa tym większe podobieństwo,
- podobieństwo pomiędzy pojęciami A i B związane jest z różnicami ich znaczeń, im one większe tym mniejsze podobieństwo,
- identyczność, czyli zawieranie tej samej informacji, pojęć A i B powoduje osiągnięcie maksymalnego podobieństwa niezależnie od ich wspólności znaczeniowej.

Opierając się na powyższych przesłankach autorzy opracowali sześć założeń:

1. Wspólność znaczeniowa pojęć A i B opisana jest wyrażeniem 4.1

$$I(common(A,B)) \tag{4.1}$$

gdzie $common(A,B)$ jest stwierdzeniem s opisującym bliskość znaczenia A i B, a $I(s)$ jest ilością informacji zawartej w tym stwierdzeniu. Zgodnie z teorią informacji [36], ilość informacji zawarta w stwierdzeniu równa jest przeciwności logarytmu z prawdopodobieństwa P jego wystąpienia (równanie 4.2).

$$I(common(A,B)) = -\log P(common(A,B)) \tag{4.2}$$

2. Różnica pomiędzy A i B liczona jest wyrażeniem 4.3,

$$I(description(A,B)) - I(common(A,B)) \tag{4.3}$$

gdzie $I(description(A,B))$ jest opisem A i B.

3. Podobieństwo $sim_{lin}(A,B)$ pojęć A i B jest funkcją ich wspólności znaczeniowej oraz ich różnicy (równanie 4.4).

$$sim_{lin}(A,B) = f(I(common(A,B)), I(description(A,B))) \tag{4.4}$$

Dziedziną funkcji f jest $\{(x,y)|x \geq 0, y > 0, y \geq x\}$. Ponadto autorzy algorytmu przyjęli, że są to jedyne czynniki wpływające na podobieństwo dwóch pojęć.

4. Podobieństwo pomiędzy dwoma identycznymi pojęciami przyjmuje stałą, maksymalną wartość niezależną od tych pojęć. Tą wartością jest 1 (wyrażenie 4.5).

$$\forall x > 0, f(x,x) = 1 \tag{4.5}$$

5. Jeżeli pomiędzy dwoma pojęciami nie ma żadnej wspólności znaczeniowej, to ich podobieństwo przyjmuje stałą, minimalną wartość niezależną od różnic pomiędzy tymi pojęciami. Tą wartością jest 0 (równanie 4.6).

$$\forall y > 0, f(0,y) = 0 \tag{4.6}$$

6. W przypadku, gdy pojęcia A i B mogą zostać porównane z dwu punktów widzenia, ich podobieństwo obliczane jest jako średnia ważona poszczególnych podobieństw uzyskanych z każdego puntu widzenia (równanie 4.7).

$$\begin{aligned} \forall x_1 \leq y_1, x_2 \leq y_2 : f(x_1 + x_2, y_1 + y_2) \\ = \frac{y_1}{y_1 + y_2} f(x_1, y_1) + \frac{y_2}{y_1 + y_2} f(x_2, y_2) \end{aligned} \tag{4.7}$$

Na podstawie powyższych założeń Lin sformułował oraz udowodnił twierdzenie 4.3.1.

Twierdzenie 4.3.1 *Podobieństwo pomiędzy pojęciami A i B opisane jest stosunkiem ilości informacji niezbędnej do opisania ich wspólności znaczeniowej oraz ilością informacji niezbędnej do ich opisania (wzór 4.8).*

$$sim_{lin}(A,B) = \frac{\log P(common(A,B))}{\log P(description(A,B))} \tag{4.8}$$

Dowód

$$\begin{aligned} f(x,y) &= f(x+0, x+(y-x)) \\ &= \frac{x}{y} * f(x,x) + \frac{y-z}{x} * f(0, y-z) \\ &= \frac{x}{y} * 1 + \frac{y-z}{x} * 0 \\ &= \frac{x}{y} \quad \blacksquare \end{aligned}$$

W taksonomii, jaką jest WordNet, podobieństwo semantyczne klas C i C' nie dotyczy samych klas, a ogólnych elementów w nich zawartych [97]. Porównując więc np. rzeki i kanały nie porównujemy zbioru rzek ze zbiorem kanałów, a ogólną rzekę z ogólnym kanałem. W związku z tym Lin zdefiniował podobieństwo $sim_{lin}(C, C')$ jako podobieństwo pomiędzy x a x', gdzie $x \in C$ oraz $x' \in C'$.

Jako że stwierdzenia $x \in C$ oraz $x' \in C'$ są niezależne, to ilość informacji zawartej w stwierdzeniu „$x \in C$ oraz $x' \in C'$" opisana jest wzorem 4.9, gdzie $P(C)$ oraz $P(C')$ jest prawdopodobieństwem przynależności losowo wybranego pojęcia odpowiednio do C i C'.

$$-\log P(C) - \log P(C') \tag{4.9}$$

Zakładając, że stosowana taksonomia posiada drzewiastą strukturę powiązań pomiędzy pojęciami, to dla $x_1 \in C_1$ oraz $x_2 \in C_2$ wspólność znaczeniowa wyraża się zapisem $x_1 \in C_0 \wedge x_2 \in C_0$, gdzie C_0 jest najbardziej szczegółową klasą nadrzędną względem zarówno C_1 jak i C_2. Podobieństwo semantyczne $sim_{lin}(x_1, x_2)$ opisuje więc równanie 4.10.

$$sim_{lin}(x_1, x_2) = \frac{2 * \log P(C_0)}{\log P(C_1) + \log P(C_2)} \tag{4.10}$$

Zaproponowana miara podobieństwa przeważa nad innymi, zwłaszcza miarami opartymi na odległości pomiędzy pojęciami, korelacją z ludzką oceną podobieństwa ([104]) jak i prostotą użycia dzięki ograniczonemu zbiorowi zwracanych wyników. Dzięki temu idealnie nadaje się do zastosowania jako podstawa analizy semantycznej pojęć zawartych w ontologiach.

Złożoność obliczeniowa miar podobieństwa opartych na WordNet zazwyczaj podana jest w liczbie wyszukiwań w jego taksonomii. Złożoność algorytmu jest więc wprost proporcjonalna do liczby znaczeń porównywanych pojęć, gdyż konieczne jest porównanie każdego ze znaczeń pierwszego słowa z każdym ze znaczeń drugiego słowa, oraz głębokością h hierarchii słownika WordNet [79]. Jeżeli liczba znaczeń pierwszego pojęcia wynosi p_1 a drugiego p_2, to złożoność porównania pojedynczej pary wyrazów wynosi $O(p_1 p_2 h)$.

4.3.2 Odległość edycyjna Levenshteina

Algorytmy oceny podobieństwa semantycznego bardzo dobrze działają w połączeniu z rozbudowanymi słownikami, będącymi dla nich swoistą bazą wiedzy, na podstawie której oceniane jest podobieństwo dwu pojęć. Słowniki takie posiadają niestety skończoną liczbę pojęć, będącą zaledwie podzbiorem tego, co oferuje dowolny z języków naturalnych. Słownik WordNet, jedna z najbardziej rozbudowanych taksonomii, zawiera obecnie 155287 różnych pojęć [121]. Istnieje więc duże prawdopodobieństwo napotkania pojęcia, które nie jest w niej zawarte. Rozwiązaniem takiej sytuacji jest obliczanie podobieństwa pomiędzy pojęciami za pomocą metody Levenshteina ([94]), gdzie podobieństwo pomiędzy wyrazami oblicza się jako funkcję najmniejszej liczby przekształceń elementarnych niezbędnych do przejścia z jednego pojęcia do drugiego.

Jako przekształcenia elementarne traktowane są:

- wstawienie znaku do ciągu opisującego pojęcie,
- usunięcie znaku z ciągu opisującego pojęcie,
- zamianę znaku w ciągu opisującym pojęcie na inny znak.

Algorytm Levenshteina określa więc tzw. odległość edycyjną (ang. *edit distance*), mówiącą ile podstawień, usunięć i wstawień należy wykonać by porównywane ciągi znaków stały się identyczne.

Przebieg algorytmu dla dwu danych ciągów znaków s oraz t opisuje Algorytm 1.

Algorytm 1 Algorytm Levenshteina [63]

1. Niech $n = \textit{długość}(s)$ a $m = \textit{długość}(t)$.
2. Jeżeli $n == 0$ zwróć m i zakończ działanie.
3. Jeżeli $m == 0$ zwróć n i zakończ działanie.
4. Utwórz macierz d o $m+1$ wierszach indeksowanych od 0 do m oraz $n+1$ kolumnach indeksowanych od 0 do n, pierwszy wiersz zainicjuj wartościami od 0 do n a pierwszą kolumnę od 0 do m.
5. Porównaj każdy znak pierwszego ciągu, indeksowany za pomocą i, z każdym znakiem drugiego ciągu, indeksowanym za pomocą j. Dla każdego porównania:
   (a) jeżeli $s[i]$ jest identyczne z $t[j]$, $koszt = 0$, w przeciwnym wypadku $koszt = 1$,
   (b) ustaw wartość komórki $d[i, j]$ macierzy na wartość minimalną spośród:
   - $d[i-1, j] + \ 1$,
   - $d[i, j-1] + \ 1$,
   - $d[i-1, j-1] + \ koszt$.
6. Odczytaj odległość $dist_{lev}(s, t)$ z komórki $d[n, m]$.

Algorytm 1 po zakończeniu porównania ciągów s i t, zwraca wartość odległości $dist_{lev}(s, t)$ należącą do przedziału $< 0, max(m, n) >$. W przypadku dwu identycznych ciągów wartość odległości wynosi 0, a w przypadku przeciwnym, gdy porównywane ciągi są różne, odległość równa będzie liczbie liter w dłuższym z nich. Stąd konieczność znormalizowania wyników do zakresu zwracanego przez algorytm Lin (Równanie 4.11).

$$sim_{lev} = 1 - \frac{dist_{lev}(s, t)}{max(\textit{długość}(s), \textit{długość}(t))} \tag{4.11}$$

Tak przygotowana miara podobieństwa może służyć jako uzupełnienie miary Lin w przypadku, gdy przynajmniej jedno z porównywanych pojęć nie ma swojego odzwierciedlenia w słowniku WordNet. Złożoność obliczeniowa tej operacji wynosi $O(mn)$, gdyż wymaga porównania każdego znaku pierwszego ciągu z każdym znakiem drugiego ciągu.

4.3.3 Wyznaczanie podobieństwa zbiorów pojęć

Algorytmy opisane w punktach 4.3.1 oraz 4.3.2 niniejszego rozdziału osiągają bardzo dobre rezultaty w przypadku porównywania pojedynczych pojęć. W przypadku zbioru pojęć, jak np. przy porównywaniu komentarzy czy wielowyrazowych nazw, konieczne jest zastosowanie dodatkowych

algorytmów znajdujących najlepszy zbiór podobieństw pomiędzy poszczególnymi pojęciami z obu zbiorów.

Problem analizy podobieństwa dwu zbiorów pojęć można sprowadzić do problemu przypisania [33], który brzmi [92]:
„Dane jest n pracowników oraz n zadań. Dana jest macierz kwalifikacji $Q = (q_{ij})$ mówiąca o tym, jak dobrze pracownik i jest w stanie wykonać zadanie j (q_{ij} = jakość). Problem przypisania polega na znalezieniu takiego przypisania pracowników do zadań, by zadania były wykonane jak najlepiej, a każdy pracownik wykonywał tylko jedno zadanie na raz."

Jednym z popularnych algorytmów rozwiązujących ten problem jest algorytm zwany Metodą Węgierską [60, 92, 105]. Działanie algorytmu najwygodniej zaprezentować definiując problem poprzez pełny graf dwudzielny $G = (W, T, E)$, gdzie W jest zbiorem wierzchołków reprezentujących n pracowników, T jest zbiorem wierzchołków reprezentujących x zadań, $W \cap T \in \varnothing$, a $E \subseteq W \times T$ jest zbiorem ważonych krawędzi łączących wszystkie elementy zbioru W ze wszystkimi elementami zbioru T [26]. Wartość wagi krawędzi $e_{ij} \in E$ równa jest jakości wykonania zadania $t_j \in T$ przez pracownika $w_i \in W$. Złożoność obliczeniowa algorytmu wynosi $O(x^3)$ [105].

Poszukiwane jest doskonałe dopasowanie (ang. *perfect matching*) o najwyższej łącznej jakości, gdzie dopasowaniem nazywamy podzbiór $M \subseteq E$ taki, że $\forall w \in W$ oraz $\forall t \in T$ co najwyżej jedna krawędź z M sąsiaduje z tymi wierzchołkami. Rozmiarem dopasowania $|M|$ nazywamy liczbę krawędzi należących do M. Doskonałym dopasowaniem nazywamy takie dopasowanie M, dla którego każde inne dopasowanie M' spełnia zależność $|M'| \leq |M|$ oraz każdy z wierzchołków grafu G sąsiaduje z krawędzią w M.

Potencjałem nazywamy funkcję $y : (W \cup T) \mapsto \mathbb{R}$, dla której prawdziwa jest zależność 4.12.

$$y(i) + y(j) \geq q_{ij} \forall i \in W, j \in T \tag{4.12}$$

Wartość potencjału y wyraża się wzorem 4.13

$$y = \sum_{v \in S \cup T} y(v) \tag{4.13}$$

Grafem równości (ang. *equality graph*) w stosunku do y nazywamy graf $G_y = (W, T, E_y)$ gdzie $E_y = \{(i, j) : y(i) + y(j) = q_{ij}\}$.

Kuhn i Munkers [92, 105] wysnuli i udowodnili twierdzenie 4.3.2, które przekształca problem z optymalizacyjnego na kombinatoryczny.

Twierdzenie 4.3.2 *Jeżeli y jest potencjałem oraz M jest doskonałym dopasowaniem w E_y to M jest dopasowaniem o najwyższej jakości.*

Dowód Niech krawędź $e \in E$ oznaczona będzie poprzez (e_i, e_j), a M' będzie dowolnym doskonałym dopasowaniem w E. Każdy wierzchołek $v \in (W \cup T)$ pokryty jest dokładnie jeden raz przez M' więc

$$q(M') = \sum_{e \in M'} q(e) \leq \sum_{e \in M'} (y(e_i) + y(e_j)) = \sum_{v \in (W \cup T)} y(v)$$

$\sum_{v \in (W \cup T)} y(v)$ jest górnym ograniczeniem kosztu każdego doskonałego dopasowania.
Niech M będzie doskonałym dopasowaniem w E_y. Uzyskujemy wtedy

$$q(M) = \sum_{e \in M} q(e) = \sum_{v \in (W \cup T)} y(v)$$

Więc $y(M') \leq y(M)$ a M jest optymalne. ∎

Algorytm 2 Metoda Węgierska [60, 92, 105]

1. Rozpocznij obliczenia z dowolnym potencjałem y i dowolnym dopasowaniem M w E_y.
2. Wykonuj następujące kroki póki M nie jest doskonałym dopasowaniem:
   (a) Rozszerz M w E_y zwiększając liczbę należących do zbioru M krawędzi.
   (b) Jeżeli nie da się rozszerzyć M, skoryguj potencjał y na y' tak, by $E_y \subset E_{y'}$.

Ogólny przebieg algorytmu prezentuje Algorytm 2.

Należy zauważyć, że każdy z kroków algorytmu zwiększa albo rozmiar M, albo rozmiar E_y, co gwarantuje jego skończoność, a zgodnie z twierdzeniem 4.3.2 uzyskane w wyniku pracy algorytmu dopasowanie M będzie doskonałym dopasowaniem o maksymalnej sumie wag.

Należy zauważyć, że algorytm dopuszcza jako punkt wyjścia dowolny potencjał i dowolne dopasowanie, np. wg równania 4.14.

$$\forall w \ \in W, y(w) = 0, \forall t \ \in T, y(t) = \max_{w \in W}\{y(t,w)\} \tag{4.14}$$

Dla takich założeń zawsze zachodzi warunek 4.15.

$$\forall w \ \in W, t \ \in T, q(x) \leq y(w) + \ y(t) \tag{4.15}$$

A jak przebiega proces modyfikacji potencjału? Przyjmijmy, że y jest potencjałem. Zdefiniujmy sąsiedztwo $N_y(\cdot)$ wierzchołka $u \in W \ \cup T$ oraz zbioru $A \subseteq W \ \cup T$ jako

$$N_y(u) = v: \ (u,v) \in E_y, N_y(A) = \bigcup_{u \in A} N_y(u)$$

Niech $A \subseteq T$, $C = N_y(A) \neq W$, a $\alpha_y = \min_{a \in A, b \notin C}\{y(a) + \ y(b) - q(a,b)\}$ oraz

$$y'(v) = \begin{cases} y(v) - \alpha_y & \text{jeżeli } v \in A \\ y(v) + \ \alpha_y & \text{jeżeli } v \in C \\ y(v) & \text{w innych przypadkach} \end{cases}$$

W wyniku tych działań uzyskane y' jest potencjałem oraz

- jeżeli krawędź $(a,b) \in E_y$, gdzie $a \in A, b \ \in C$ to $(a,b) \in E_{y'}$,
- jeżeli krawędź $(a,b) \in E_y$, gdzie $a \notin A, b \ \notin C$ to $(a,b) \in E_{y'}$,
- istnieje krawędź $(a,b) \in E_{y'}$ dla $a \in A, b \ \notin C$.

Powyższa reprezentacja jest jedną z wielu dla tego algorytmu, inną popularną jest reprezentacja macierzowa opisana w [105].

Problem znajdowania podobieństwa pomiędzy zbiorami pojęć jest uogólnieniem problemu przypisania. Można go zdefiniować następująco:
„Dane są dwa zbiory pojęć – z_1 o rozmiarze n oraz z_2 o rozmiarze m. Dana jest macierz kwalifikacji $Q = (q_{ij})$ mówiąca o podobieństwie pomiędzy pojęciem i zbioru z_1 a pojęciem j zbioru z_2 ($q_{ij} = sim(z_1[i], z_2[j])$). Macierz Q opisuje zbiór z_p możliwych powiązań pomiędzy elementami zbiorów z_1 i z_2. Należy znaleźć podzbiór zbioru z_p opisujący maksymalne sumaryczne podobieństwo pomiędzy elementami zbiorów z_1 i z_2, przy czym każdy z elementów zbioru z_1 może być powiązany z co najwyżej jednym elementem zbioru z_2 oraz każdy z elementów zbioru z_2 może być

powiązany z co najwyżej jednym elementem zbioru z_1. Innymi słowy przy wyznaczaniu sumarycznego podobieństwa każdy z wyrazów może być uwzględniony co najwyżej jeden raz. W związku z tym, przy braku gwarancji identyczności rozmiarów obu zbiorów, by zachować zgodność z oryginalnym problemem konieczne jest dopasowywanie elementów mniejszego zbioru (czyli wyrazów krótszego zdania) do elementów dłuższego zbioru (czyli wyrazów dłuższego zdania)."

Wynikiem działania algorytmu węgierskiego jest liczba będąca sumą podobieństw pomiędzy wyrazami wchodzącymi w skład najlepszego dopasowania. Dla $n \leq m$ (drugi zbiór słów posiada więcej elementów niż pierwszy) może ona przyjmować dowolną wartość z przedziału $< 0, n >$. Konieczna jest więc, podobnie jak w przypadku algorytmu Levenshteina, normalizacja do przedziału $< 0, 1 >$. W tym celu uzyskany wynik dzielony jest przez m, czyli rozmiar większego ze zbiorów pojęć.

4.4 Stopnie analizy podobieństwa elementów ontologii

Integracja wiedzy zapisanej w postaci dwu ontologii wymaga umiejętności porównywania ich elementów składowych. Nieodzowne stają się więc algorytmy określania semantycznego i syntaktycznego podobieństwa konceptów. Stoją one u podstaw wszelkich algorytmów łączenia i odwzorowywania ontologii.

Zaproponowane w tej rozprawie miary podobieństwa pomiędzy konceptami w dużej mierze wynikają bezpośrednio z pragmatycznego spojrzenia na ontologie, jakie zaproponowali zarówno Hovy [75] jak i Euzenat i Volchev [53]. Poniższa propozycja jest rozszerzeniem podejścia prezentowanego przez tych badaczy o wykorzystanie możliwości drzemiących w nowoczesnym słowniku semantycznym jaki jest WordNet. Wyróżnić możemy więc cztery miary podobieństwa wzajemnie dopełniające się i uzupełniające:

1. Podobieństwo leksykalne P_{lex} klas $\mathcal{K}_i (i = 1, 2)$ i bytów $\mathcal{B}_i (i = 1, 2)$

   Podobieństwo leksykalne jest bazową formą podobieństwa pomiędzy konceptami. Do porównywania elementów ontologii używany jest semantyczny słownik WordNet. Dzięki zdefiniowanym w nim relacjom nadrzędności i podrzędności pomiędzy słowami możliwe jest określenie wzajemnego zawierania się znaczeń porównywanych konceptów. Porównując dwa koncepty ze sobą na podstawie leksykalnego podobieństwa opisujących ich etykiet można określić ich wzajemną relację – czy są sobie równoważne, rozłączne lub jeden z nich zawiera się logicznie w drugim, czyli ustalić hierarchię dziedziczenia klas i przynależności bytów opisanych danymi konceptami. Zastosowanie słownika WordNet pozwala na bezpośrednie dopasowanie ciągów znakowych, jak i dopasowanie do synonimów poprzez operacje na synsetach, a nie samych słowach.

   Złożoność obliczeniowa tej miary, wyrażona w liczbie wyszukiwań w taksonomii słownika WordNet, wynosi $O(p_1 p_2 h \ + log(h))$, gdzie h jest głębokością hierarchii słownika WordNet, p_1 liczbą znaczeń pierwszego pojęcia, a p_2 liczbą znaczeń drugiego pojęcia. Komponent logarytmiczny wynika z określenia ścieżki przejścia z jednego pojęcia do drugiego. Sumę tę można uprościć do postaci $O(p_1 p_2 h)$.

2. Podobieństwo semantyczne P_{sem} klas $\mathcal{K}_i (i = 1, 2)$ i bytów $\mathcal{B}_i (i = 1, 2)$

   W przypadku, gdy określenie podobieństwa leksykalnego jest niemożliwe, proponowany algorytm wspiera swoje działanie za pomocą podobieństwa semantycznego. Na podstawie wartości podobieństwa semantycznego będzie można określić, czy pojęcia te są tożsame, zawierające się, czy rozłączne. Porównywane są etykiety słów i na podstawie skali podobieństwa, reprezentowanej przez liczbę z przedziału od 0 do 1, określone zostanie podobieństwo pojęć. W tym celu stosowany będzie opisany w rozdziale 4.3.1 niniejszej pracy algorytm Lin [96, 97]. Algorytm Lin opiera się na WordNet, dlatego do jego poprawnej pracy konieczna jest obecność poszukiwanych pojęć w tym słowniku. Niestety nie zawsze warunek ten jest

spełniony. W takiej sytuacji, jako algorytm pomocniczy, stosowana jest odległość edycyjna Levenshteina. Metoda ta została opisana w rozdziale 4.3.2. Techniką tą można określić jedynie czy analizowane pojęcia są tożsame czy różne. Nie ma możliwości określenia wzajemnej ogólności reprezentowanych przez pojęcia konceptów. Ponadto, w celu poprawy wydajności i eliminacji zwrotów nie posiadających znaczenia (jak np. zaimki, rodzajniki itp.), porównywane są jedynie te ciągi znakowe, które są dłuższe niż dwa znaki. W słowniku WordNet jest zaledwie 305 rzeczowników składających się z dwu lub mniej znaków, co stanowi 0,26% wszystkich zawartych tam rzeczowników. W większości są to skróty i odpowiedniki liczb w systemie rzymskim. Ograniczenie takie nie zmniejsza więc ogólności przyjętej metody.

Złożoność obliczeniowa tej miary, wyrażona częściowo w liczbie wyszukiwań w taksonomii słownika WordNet, a częściowo w rozmiarze słów porównywanych poprzez algorytm Levenshteina, wynosi $O(p_1 p_2 h + mn)$, gdzie h jest głębokością hierarchii słownika WordNet, p_1 liczbą znaczeń pierwszego pojęcia, p_2 liczbą znaczeń drugiego pojęcia, m liczbą znaków w etykiecie pierwszego pojęcia, a n liczbą znaków w etykiecie drugiego pojęcia.

Tablica 4.2: Zestawienie podobieństwa par testowych Millera i Charlesa [104] wg ludzi i wg miary Lin na podstawie słownika WordNet

Wyraz 1	Wyraz 2	Odsetek grupy testowej skłonny uznać pojęcia za identyczne	Podobieństwo wg ludzi	Podobieństwo wg miary Lin
car	automobile	100,00%	0,92	1,00
gem	jewel	82,61%	0,78	1,00
journey	voyage	100,00%	0,90	0,80
boy	lad	82,61%	0,77	0,92
coast	shore	95,65%	0,85	0,97
asylum	madhouse	69,57%	0,68	0,86
magician	wizard	82,61%	0,84	1,00
midday	noon	82,61%	0,81	1,00
furnace	stove	69,57%	0,73	0,25
food	fruit	30,43%	0,46	0,18
bird	cock	34,78%	0,40	0,74
bird	crane	26,09%	0,37	0,72
tool	implement	56,52%	0,54	0,95
brother	monk	34,78%	0,42	0,97
crane	implement	8,70%	0,26	0,38
lad	brother	26,09%	0,29	0,25
journey	car	4,35%	0,16	0,00
monk	oracle	4,35%	0,22	0,22
cemetery	woodland	0,00%	0,09	0,13
food	rooster	17,39%	0,25	0,09
coast	hill	0,00%	0,16	0,64
forest	graveyard	0,00%	0,09	0,13
shore	woodland	0,00%	0,12	0,14
monk	slave	4,35%	0,09	0,24
coast	forest	0,00%	0,11	0,14
lad	wizard	0,00%	0,07	0,25
chord	smile	4,35%	0,07	0,28
glass	magician	0,00%	0,07	0,21
noon	string	0,00%	0,02	0,07
rooster	voyage	0,00%	0,02	0,00

Poziom podobieństwa stanowiący o identyczności elementów określony został na podstawie

przeprowadzonych badań. Grupie 23 zróżnicowanych pod względem wieku i miejsca zatrudnienia osób przedstawiono 30 par wyrazów zdefiniowanych przez Millera i Charlesa [104]. Zadaniem tej grupy było określenie, czy wyrazy zgrupowane w pary są wg nich wystarczająco podobne by uznać je za identyczne, czy są rozbieżne. Uzyskane rezultaty zestawiono z subiektywnym odczuciem podobieństwa porównywanych wyrazów oraz podobieństwem określonym za pomocą algorytmu opracowanego przez Lin. Wyniki tych badań przedstawiono w Tablicy 4.2. Pary wyrazów o wysokim stopniu podobieństwa wg Lin (czyli: *car* - *automobile*, *gem* - *jewel*, *journey* - *voyage*, *boy* - *lad*, *coast* - *shore*, *asylum* - *madhouse*, *magician* - *wizard*, *midday* - *noon* oraz *furnace* - *stove*) zostały uznane przez ludzi za bardzo podobne, w wyniku czego zasadne jest ich połączenie. Podobieństwo par: *tool* - *implement* oraz *brother* - *monk*, pomimo wysokiego współczynnika podobieństwa semantycznego wg Lin, zostało nisko ocenione przez ludzi. Wynikało to w dużej mierze z nieznajomości dodatkowych znaczeń słów *brother* oraz *implement*. Osoby związane bliżej z językiem angielskim (osoby przebywające na stałe w Irlandii Północnej, nauczycielka języka angielskiego) oceniły podobieństwo pomiędzy tymi wyrazami jako wysokie. Największym problemem okazały się wyrazy o znaczeniach zawierających się w sobie: *bird* - *cock* oraz *bird* - *crane*. W obu przypadkach drugi z wyrazów znaczeniowo zawiera się w pierwszym. Mając do dyspozycji jedynie liczbową wartość podobieństwa nie da się określić, który z wyrazów ma szersze znaczenie, a który węższe. Wg miary podobieństwa Lin wyrazy te mają jednak dość duże podobieństwo semantyczne (odpowiednio 0,74 i 0,72). Część osób uznała je jako wystarczająco podobne, by połączyć opisane nimi koncepty w jeden, zakładając, że znana jest tylko wartość podobieństwa między nimi. Pozostałe wyrazy zarówno przez ludzi jak i miarę podobieństwa opracowaną przez Lin uznane zostały za różne. Pozyskane obserwacje są zbieżne z wynikami uzyskanymi przez innych badaczy, np. grupę odpowiedzialną za narzędzie Falcon-AO [81]. Na podstawie uzyskanych od grupy testowej wyników oraz badań innych badaczy stwierdzono więc, że dla $P_{sem} < 0,7$ badane koncepty są uznawane za różne, a $P_{sem} \geq 0,7$ za identyczne.

3. Podobieństwo komentarzy P_{kom} przypisanych do klas $\mathcal{K}_i (i = 1, 2)$ i bytów $\mathcal{B}_i (i = 1, 2)$

   Kolejnym atrybutem pozwalającym określić podobieństwo dwóch elementów ontologii jest tożsamość komentarza jednego z elementów z komentarzem drugiego. Podejście to traktuje komentarze jako zbiory elementów grafu dwudzielnego, którego krawędzie oznaczają podobieństwo pomiędzy składowymi obu komentarzy. Podobieństwo pomiędzy oboma ciągami wyrazów wyraża się więc poprzez maksymalne dopasowanie w tak utworzonym grafie dwudzielnym. Ocena podobieństwa oparta na komentarzach elementów sprowadza się więc do następującego problemu:

   „Dany jest graf $\mathcal{G}(V, E)$, gdzie V jest zbiorem wierzchołków grafu, a E zbiorem krawędzi grafu. $\mathcal{G}$ może być podzielony na dwa rozłączne zbiory wierzchołków L oraz R takich, że każda krawędź $e \in E$ w grafie $\mathcal{G}$ łączy wierzchołek należący do L z wierzchołkiem należącym do R i posiada nieujemną wagę. Zbiór L jest zbiorem elementów pierwszego komentarza, a zbiór R jest zbiorem elementów drugiego komentarza. Wartość wag krawędzi $e \in E$, łączącej wierzchołek $l \in L$ z wierzchołkiem $r \in R$, równa jest podobieństwu P_{ek} pomiędzy elementem l i elementem r. Podobieństwo $P_{ek_{lr}}$ tożsame jest z podobieństwem semantycznym pomiędzy elementem l i elementem r, które wyznaczane jest identycznie jak podobieństwo P_{sem} opisane w punkcie 2 niniejszego podrozdziału. Podobnie jak w przypadku P_{sem} w pierwszej kolejności podobieństwo pomiędzy elementami l i r określane jest na podstawie algorytmu Lin i słownika WordNet. Jeżeli któreś z tych pojęć nie ma swojego odzwierciedlenia w tym słowniku, stosowany jest algorytm Levenshteina."

   Porównanie komentarzy analizowanych elementów sprowadza się więc do znalezienia podzbioru krawędzi łączących wszystkie elementy zbiorów L i R o najwyższej sumie wag. Zadanie to zostało zrealizowane metodą węgierską [92].

   Wartość podobieństwa P_{kom} wyznaczana jest jako stosunek sumy wartości wag przypisanych do elementów uzyskanego zbioru krawędzi do liczby elementów w dłuższym komentarzu (wzór 4.16).

$$P_{kom} = \frac{\sum_i P_{ek_i}}{max(|L|, |R|)} \tag{4.16}$$

Podejście takie zapewnia odpowiedni przedział uzyskiwanych wartości podobieństwa (z przedziału $< 0, 1 >$) oraz uwzględnia długość porównywanych komentarzy.

Złożoność obliczeniowa tej miary wynika ze złożoności metody węgierskiej oraz metody określania podobieństwa elementów składowych zbiorów pojęć, jakimi są komentarze. Złożoność obliczeniowa metody węgierskiej wynosi $O(x^3)$, gdzie x to liczba wyrazów w każdym z komentarzy. Inicjacja tablicy podobieństw wymaga wykonania x^2 określeń podobieństwa, gdzie złożoność każdego z nich wynosi $O(p_1 p_2 h\ + mn)$, gdzie h jest głębokością hierarchii słownika WordNet, p_1 liczbą znaczeń pierwszego pojęcia, p_2 liczbą znaczeń drugiego pojęcia, m liczbą znaków w etykiecie pierwszego pojęcia, a n liczbą znaków w etykiecie drugiego pojęcia. Całkowita złożoność obliczeniowa tej miary wynosi więc $O(x^2(p_1 p_2 h\ + mn) +\ x^3)$

4. Podobieństwo strukturalne P_{str} klas $\mathcal{K}_i (i = 1, 2)$ i bytów $\mathcal{B}_i (i = 1, 2)$

   Opisuje ono w jakim stopniu badane elementy są osadzone wśród innych je otaczających. Istotny tutaj jest rodzaj, kierunek oraz ilość relacji w jakich uczestniczą oba elementy. Podobieństwo te można wyrazić wzorem:

$$P_{str} = \frac{\sum_i min(r_i)}{\sum_i max(r_i)} \tag{4.17}$$

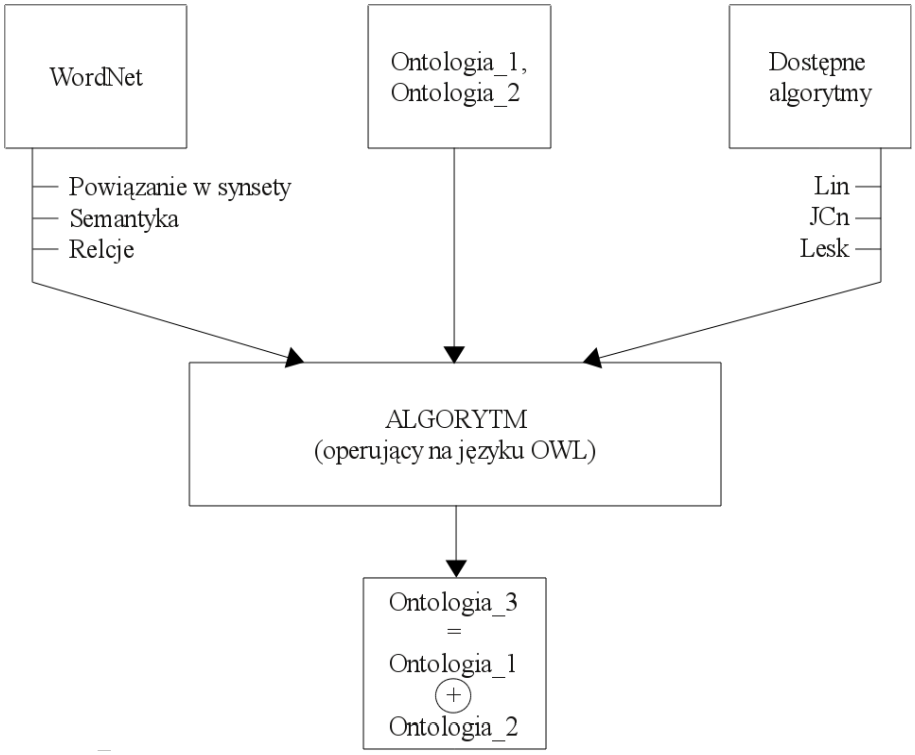
   gdzie: r_i – ilość wystąpień i-tej relacji (i = dziedziczenie, przynależność, tożsamość, rozłączność, unia, przecięcie).

   Podobieństwo to gra szczególną rolę przy rozwiązywaniu konfliktów. Większe podobieństwo strukturalne umożliwia rozróżnienie elementów podobnych leksykalnie na podstawie związków z innymi elementami ontologii, czy też posiadanych atrybutach. Liczba operacji niezbędna do wyznaczenia tej miary uzależniona jest od ilości r relacji powiązanych z analizowanymi pojęciami i wynosi $O(r)$.

Bazową miarą podobieństwa w proponowanym podejściu jest podobieństwo leksykalne. Wraz z rozwojem słowników semantycznych i zawieraniu w nich coraz to większej liczby pojęć podejście to staje się coraz bardziej uzasadnione. W wersji 3.0 słownika WordNet obecnych jest 155287 różnych pojęć i 206941 par wiążących słowo z jego znaczeniem [121]. Wysoce prawdopodobne jest więc znalezienie bezpośredniego dopasowania dla słów użytych do określenia konceptów w ontologii. W takiej sytuacji zależności pomiędzy porównywanymi konceptami odczytane zostaną z hierarchii zawartej w słowniku WordNet i na tej podstawie elementy te zostaną ulokowane w wynikowej ontologii. Kolejne dwie proponowane miary podobieństwa wykorzystane zostaną w przypadku, gdzie przynajmniej jeden z konceptów nie będzie miał bezpośredniego odwzorowania w słowniku WordNet. W takim przypadku zależności pomiędzy porównywanymi elementami określone zostaną na podstawie semantycznego podobieństwa ich etykiet. W przypadku, gdy wyznaczenie podobieństwa leksykalnego i semantycznego okaże się niemożliwe, miara podobieństwa określona zostanie na podstawie strukturalnego podobieństwa konceptów. O podobieństwie zadecyduje tutaj liczba i rodzaj relacji, w jakich ujęte są oba porównywane elementy ontologii.

4.5 Proponowany algorytm odwzorowywania i łączenia ontologii

Opisane we wcześniejszych rozdziałach właściwości słownika WordNet umożliwiają opracowanie mechanizmu odwzorowywania i łączenia ontologii. Zaobserwowano ponadto, że wiele ontologii pojęcia reprezentujące koncepty wyraża w postaci rzeczowników, a większość informacji zawartych w ontologii wyrażona jest za pomocą klas i relacji dziedziczenia zachodzących pomiędzy nimi. Dziedzinowość z kolei zwiększa prawdopodobieństwo występowania wspólnej bazy koncepcyjnej służącej do opisu danego wycinka świata [24]. Dzięki temu proponowane podejście, przede wszystkim na podstawie związków między słowami zdefiniowanymi w słowniku WordNet, ale także dzięki leksykalnej i strukturalnej analizie łączonych ontologii, umożliwia dynamiczne tworzenie zintegrowanej ontologii łączącej wiedzę zawartą w ontologiach wejściowych [19].



Rysunek 4.2: Konstrukcja wynikowej ontologii

Słownik WordNet zaproponowano więc jako bazę pojęciową umożliwiającą sprawne określanie relacji między zastosowanymi w łączonych ontologiach opisami konceptów. Na rys. 4.2 przedstawiono elementy wejściowe i wyjściowe proponowanego algorytmu. Oprócz WordNet, proponowany algorytm korzysta z algorytmu określającego podobieństwo słów opracowanego przez Lin [96, 97] wsparty przez algorytm wyznaczania odległości edycyjnej Levenshteina [94], a w przypadku dopasowania elementów wielowyrazowych stosowana jest metoda węgierska [92]. Na wejście przyjmuje dwie ontologie w języku OWL, które mają zostać połączone. Wynikiem pracy proponowanego algorytmu jest trzecia ontologia, również zapisana w języku OWL, będąca połączeniem ontologii wejściowych.

Rozpoczynając pracę algorytm pobiera na wejście dwie ontologie wejściowe A i B (wiersze 42 oraz 43). Wyjściem algorytmu jest ontologia wynikowa C, tworzona w trakcie jego pracy (wiersz 44) oraz zwracana po jego zakończeniu (wiersz 49). Bazując na założeniu, że w ontologiach opisanych w języku OWL istnieje zawsze wspólny korzeń *owl:Thing* jednakowy dla wszystkich konceptów, pobieramy go dla każdej z ontologii (wiersze 45-47) i przeprowadzamy operację łączenia poddrzew

Algorytm 3 Pseudokod prezentujący pracę algorytmu łączenia ontologii

```
1: program OntologyMerger {
2: function combineSubTrees(Node root_A, Node root_B, Node root_C) {
3: for all child_A : root_A.getChildren() do
4:   for all child_B : root_B.getChildren() do
5:     int result = compareConcepts(child_A, child_B);
6:     if result == EQUAL then
7:       Node node = combineConcepts(child_A, child_B);
8:       root_C.addChild(node);
9:       combineSubTrees(child_A, child_B, node);
10:    else if result == DISJOINT then
11:      root_C.addChildWithSubTree(child_A);
12:      root_C.addChildWithSubTree(child_B);
13:    else if result == A_MORE_GENERAL_THAN_B then
14:      root_C.addChild(child_A);
15:      placeNodeInSubTree(child_B, child_A);
16:    else if else if (result == B_MORE_GENERAL_THAN_A then
17:      root_C.addChild(child_B);
18:      placeNodeInSubTree(child_A, child_B);
19:    end if
20:  end for
21: end for
22: }
23: function placeNodeInSubTree(Node node, Node root) {
24: for all child : root.getChildren() do
25:   int result = compareConcepts(node, child);
26:   if result == EQUAL then
27:     Node node = combineConcepts(node, child);
28:     root_C.addChild(node);
29:     combineSubTrees(node, child);
30:   else if result == DISJOINT then
31:     root_C.addChildWithSubTree(child);
32:     root_C.addChildWithSubTree(node);
33:   else if result == NODE_MORE_GENERAL_THAN_CHILD then
34:     root_C.addChild(node);
35:     placeNodeInSubTree(child, node);
36:   else if result == CHILD_MORE_GENERAL_THAN_NODE then
37:     root_C.addChild(child);
38:     placeNodeInSubTree(node, child);
39:   end if
40: end for
41: }
42: input OWLOntology Ontology_A;
43: input OWLOntology Ontology_B;
44: output OWLOntology Ontology_C;
45: Node root_A = Ontology_A.getOWLThing();
46: Node root_B = Ontology_B.getOWLThing();
47: Node root_C = Ontology_C.getOWLThing();
48: combineSubTrees(root_A, root_B, root_C);
49: return  Ontology_C;
50: }
```

węzła *owl:Thing* ontologii A z węzłem *owl:Thing* ontologii B, a całość zapisujemy jako poddrzewo węzła *owl:Thing* ontologii C (wiersz 48). Algorytm opiera się na dwóch funkcjach:

- łączącą poddrzewa wskazanych węzłów (wiersze 2-22),
- umieszczającą węzeł jednej ontologii w drzewie drugiej ontologii tak, by jego bezpośrednim rodzicem był węzeł zawierający element uogólniający, a potomkami węzły bardziej szczegółowe – o ile takie istnieją (wiersze 23-42).

Obie te funkcje są bardzo podobne w działaniu, szczegółowo omówiona zostanie pierwsza z nich. Jako parametry wejściowe przyjmowane są trzy wartości:

- węzeł będący korzeniem poddrzewa ontologii A,
- węzeł będący korzeniem poddrzewa ontologii B,
- węzeł ontologii C, do której ma być podpięty wynik złączenia drzew będących pozostałymi parametrami funkcji.

Algorytm 3 porównuje elementy znajdujące się na tym samym poziomie w strukturze drzewa na zasadzie każdy z każdym (pętle zaczynające się w wierszach 3 i 4). W zależności od wyniku tego porównania wykonywana jest jedna z następujących akcji:

1. w przypadku, gdy pojęcia są identyczne (linie 6-10), tworzone jest pojęcie wspólne lub łączone elementy wiązane są relacją równoważności (linia 7), dodawane do ontologii wyjściowej (linia 8) a następnie poddrzewa wyznaczone przez te dwa węzły są ze sobą łączone poprzez rekurencyjne wywołanie funkcji (linia 9),
2. w przypadku, gdy pojęcia są całkowicie rozłączne (linie 10-13), są one wraz ze swoimi poddrzewami w całości przepisywane do ontologii wynikowej,
3. w przypadku, gdy obecnie porównywany węzeł z ontologii A jest bardziej ogólny w swoim znaczeniu niż węzeł ontologii B (linie 13-16), to węzeł ontologii A jest dodawany do ontologii wynikowej (linia 14), a węzeł ontologii B jest lokowany w drzewie o korzeniu będącym analizowanym węzłem ontologii A (linia 15). Lokowanie to odbywa się za pomocą bliźniaczej funkcji *placeNodeInSubTree*;
4. w przypadku, gdy obecnie porównywany węzeł z ontologii A jest bardziej szczegółowy w swoim znaczeniu niż węzeł ontologii B (linie 16-19), operacje wykonywane są analogiczne do punktu 3, jednak tym razem do ontologii wyjściowej dopisywany jest węzeł ontologii B a lokowany jest węzeł ontologii A.

Druga z funkcji w swoim działaniu jest bardzo podobna. Zamiast łączyć dwa drzewa znajduje najlepsze położenie dla wskazanego węzła jednej ontologii w poddrzewie drugiej ontologii. Różni się rodzajem oraz liczbą pobieranych parametrów wejściowych:

- węzeł, który ma być ulokowany w poddrzewie,
- węzeł spinający drzewo, w którym ulokowany ma być węzeł będący pierwszym parametrem funkcji.

Kluczowe znaczenie ma tutaj procedura porównywania elementów ontologii (*compareConcepts(child_A, child_B)*, linie 5 oraz 25). Mogą one być porównywane na cztery sposoby: leksykalnie, semantycznie, strukturalnie oraz na podstawie podobieństwa ich komentarzy. Funkcja ta w pierwszej kolejności bada podobieństwo leksykalne pomiędzy konceptami pozwalające szczegółowo określić rodzaj zależności pomiędzy pojęciami. W drugiej kolejności badane jest podobieństwo semantyczne pozwalające na określenie, czy koncepty są podobne, czy rozłączne.

W przypadku, gdy badanie podobieństwa leksykalnego i semantycznego nie powiedzie się, możliwe jest rozstrzygnięcie identyczności lub rozłączności konceptów na postawie równania:

$$P_{sk} = w_1 P_{str} + w_2 P_{kom} \tag{4.18}$$

gdzie:
P_{str} – podobieństwo strukturalne wynikające z podobieństwa typów i liczności relacji, w jakich znajdują się porównywane elementy (wzór 4.17),
P_{kom} – podobieństwo semantyczne komentarzy przypisanych do porównywanych elementów (wzór 4.16),
w_i – wagi nadane poszczególnym podobieństwom dobrane na podstawie przeprowadzonych badań, wyjściowo przyjmują one wartości: $w_1 = 0,3$; $w_2 = 0,7$.

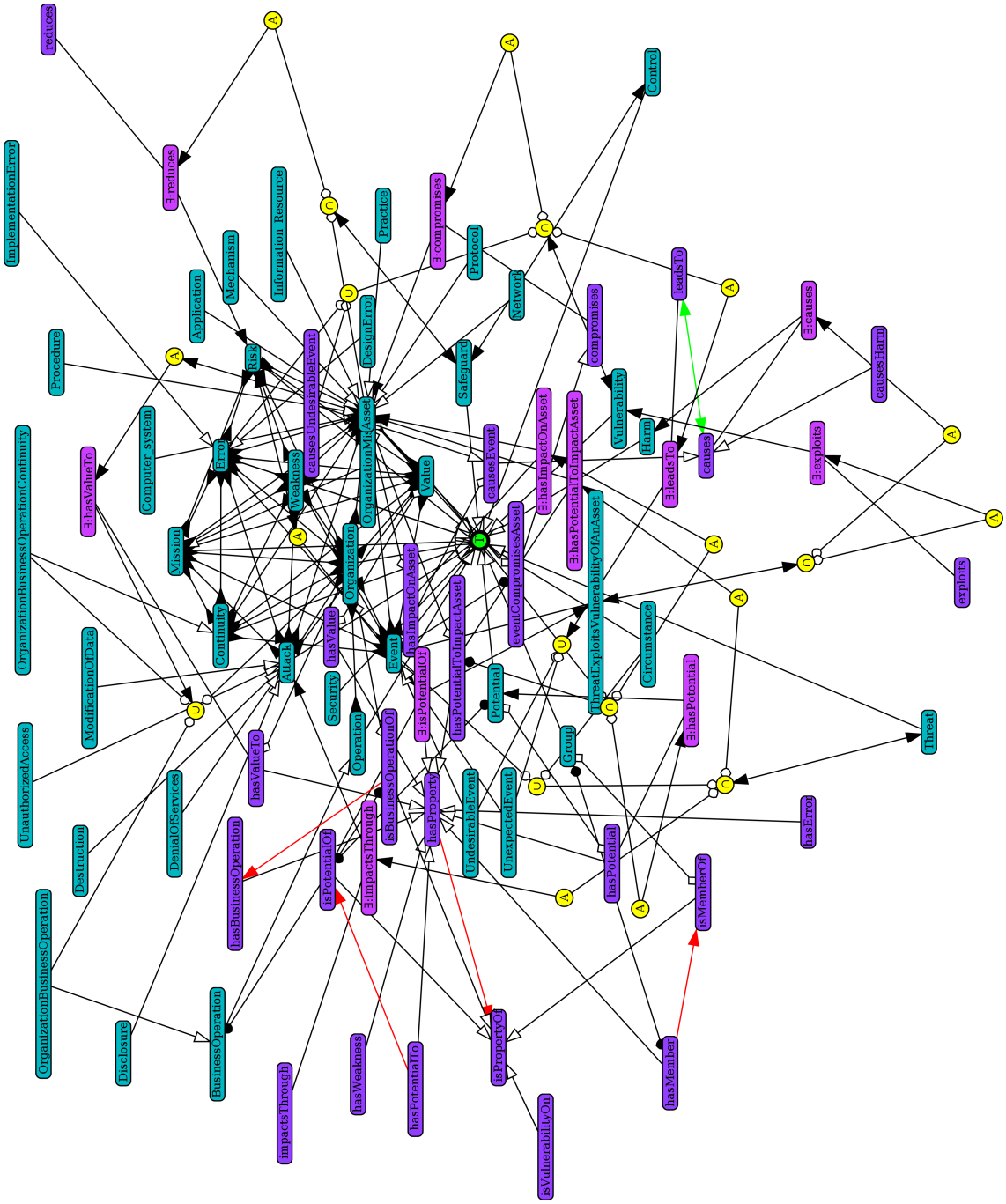
Podobnie jak w przypadku podobieństwa semantycznego, przyjęto za wartość graniczną miarę podobieństwa P_{sk} równą 0,7. $P_{sk} \geq 0,7$ oznacza elementy identyczne, a $P_{sk} < 0,7$ elementy rozłączne.

Zaproponowany w niniejszej rozprawie algorytm (Alg. 3) jest algorytmem rekurencyjnym o pesymistycznej liczbie porównań wynoszącej $O(ab)$, gdzie a to rozmiar, czyli liczba klas i bytów, ontologii wejściowej A, b to rozmiar ontologii wejściowej B. Każde porównanie dwu pojęć wymaga zastosowania zaproponowanych miar podobieństwa opisanych w części 4.4 niniejszej rozprawy. Całkowita złożoność obliczeniowa algorytmu wynosi więc $O(ab(x^2(p_1 p_2 h + mn) + x^3 + r))$, gdzie h jest głębokością hierarchii słownika WordNet, p_1 liczbą znaczeń pierwszego pojęcia, p_2 liczbą znaczeń drugiego pojęcia, m liczbą znaków w etykiecie pierwszego pojęcia, n liczbą znaków w etykiecie drugiego pojęcia, x liczbą elementów komentarzy, klas i bytów, a r ilością relacji, z jakimi powiązane są analizowane elementy.

4.6 Przykład działania algorytmu

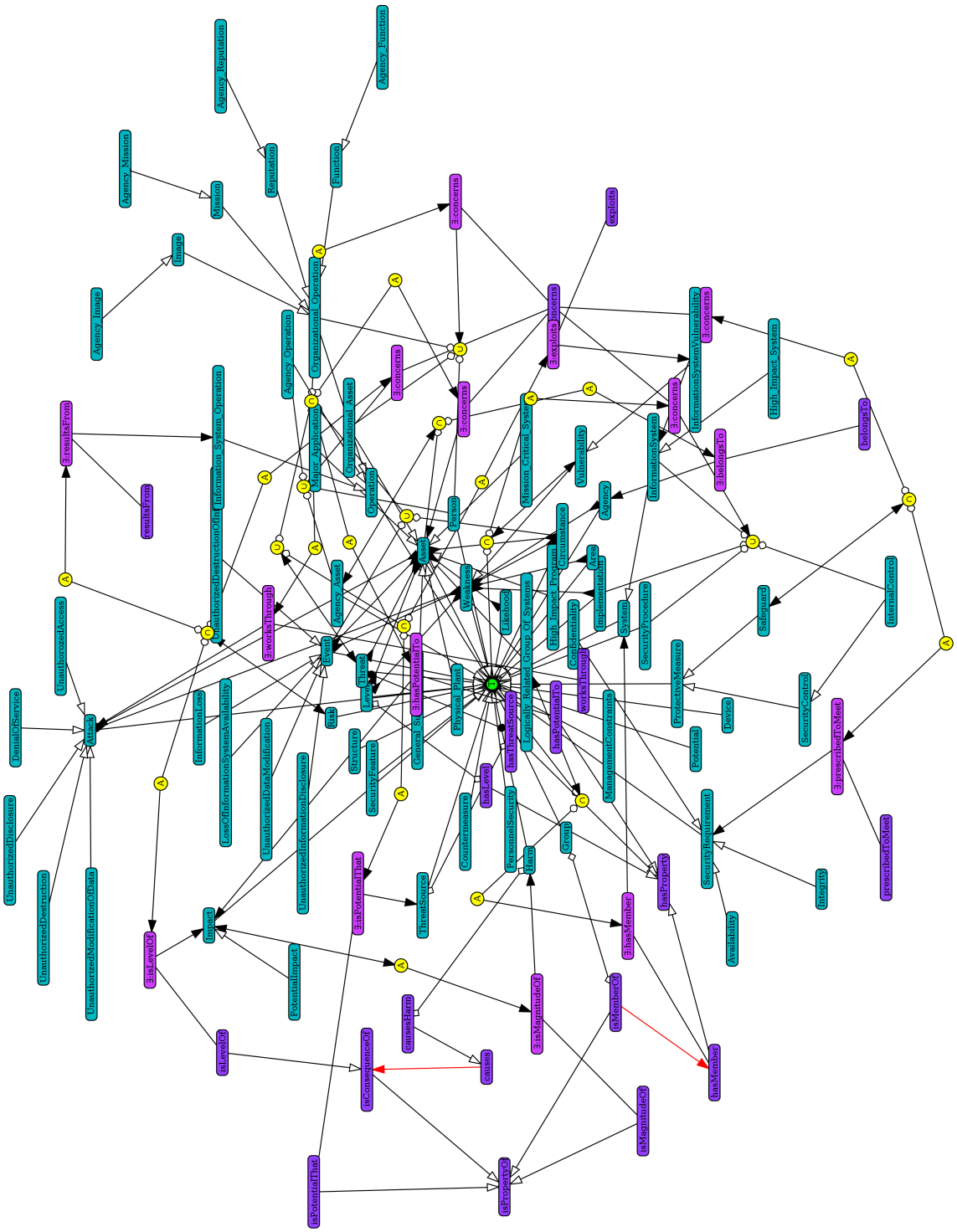
Działanie algorytmu zaprezentowane zostanie na przykładzie łączenia dwu ontologii – opartej na słowniku przygotowanym przez National Institute of Standards and Technology (NIST) [88] oraz opartej na słowniku opracowanym przez European Network and Information Security Agency (ENISA) [50]. Ontologie przygotowano korzystając ze zmodyfikowanej metodologii opracowanej przez Noy i McGuiness [24]. Dokładny opis procesu wytwarzania tych ontologii oraz zastosowanej metodologii przedstawiono w dalszej części niniejszej rozprawy w rozdziale 6.3. Obie ontologie zaprezentowane są na rys. 4.3 (ENISA) oraz rys. 4.4 (NIST), a ich wywnioskowane hierarchie odpowiednio na rys. 4.5 oraz rys. 4.6. Wizualizację wykonano za pomocą narzędzia SOVA [21]. Dla przejrzystości przykładu, prezentowane ontologie składają się jedynie z klas. Operacje łączenia bytów przeprowadzane są analogicznie, jak w przypadku klas, z tą różnicą, że byty, będąc skonkretyzowaną instancją klasy, nie mogą przynależeć do innych bytów, a jedynie do klas. Nie jest więc tworzona struktura hierarchiczna pomiędzy bytami. Do określania podobieństwa semantycznego pomiędzy pojęciami wykorzystywany jest algorytm opracowany przez Lin (opisany w rozdziale 4.3.1), wspierany przez algorytm Levenshteina (opisany w rozdziale 4.3.2) oraz Metodę Węgierską (opisana w rozdziale 4.3.3).

Algorytm rozpoczyna pracę od porównywania bezpośrednich potomków węzłów *owl:Thing* obu ontologii. Kolejność porównywania elementów zależna jest od sposobu ich odczytu w ontologii, jednak nie ma znaczenia z punktu widzenia działania algorytmu. Dla przejrzystości przyjmijmy, że elementy przeglądane są w kolejności występowania (z góry na dół) na wizualizacjach hierarchii wywnioskowanych dla przykładowych ontologii (rys. 4.5 i rys. 4.6). W pierwszej kolejności algorytm porówna więc koncept *Organization* z ontologii ENISA z konceptem *Structure* z ontologii NIST. Koncept *Organization* opatrzony jest komentarzem „an organized body of people with a particular purpose, e.g. a business (http://www.askoxford.com/concise_oed/orga-nization?view=uk)" i nie posiada klas potomnych. Koncept *Structure* nie posiada zarówno komentarza jak i klas



Rysunek 4.3: Ontologia bezpieczeństwa utworzona na bazie słownika ENISA – pełna ontologia

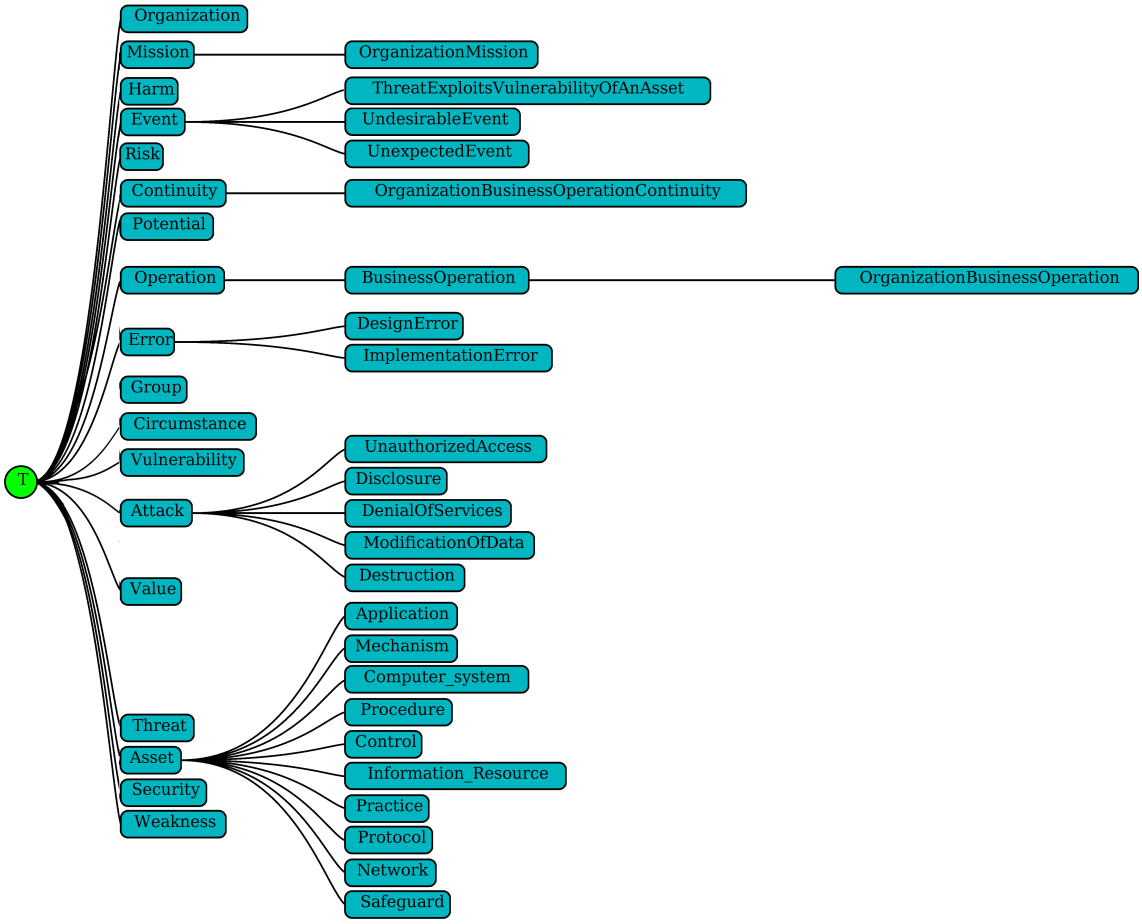
potomnych. Obie klasy są klasami potomnymi jednej klasy nadrzędnej. Dla przejrzystości, analizowane koncepty oznaczmy jako *ENISA:Organization* oraz *NIST:Structure*. W dalszej części niniejszego rozdziału koncepty pochodzące z ontologii ENISA oznaczone zostaną przedrostkiem *ENISA:* a te z ontologii NIST przedrostkiem *NIST:*. Miara podobieństwa leksykalnego w tym przypadku zwraca wartość $P_{lex} = 0$ – wyrazy będące etykietami konceptów są odmiennymi wyrazami nie będącymi synonimami. Podobieństwo semantyczne pomiędzy wyrazami wg algorytmu Lin wynosi $P_{sem} = 0.98$. Są więc one semantycznie podobne. Analizowane pojęcia w strukturze słownika WordNet związane są dodatkowo relacją podrzędności – pojęcie *Structure* jest nadrzędne



Rysunek 4.4: Ontologia bezpieczeństwa utworzona na bazie słownika NIST – pełna ontologia

względem *Organization*. W związku z tym pojęcia te zostaną dodane do wynikowej ontologii oraz związane relacją podrzędności.

Kolejne porównanie dotyczyć będzie konceptu *NIST:Group*. W tym wypadku $P_{lex} = 0$, a $P_{sem} = 0,87$. Pojęcia te są więc podobne. Analiza struktury słownika WordNet wskazuje, że kon-

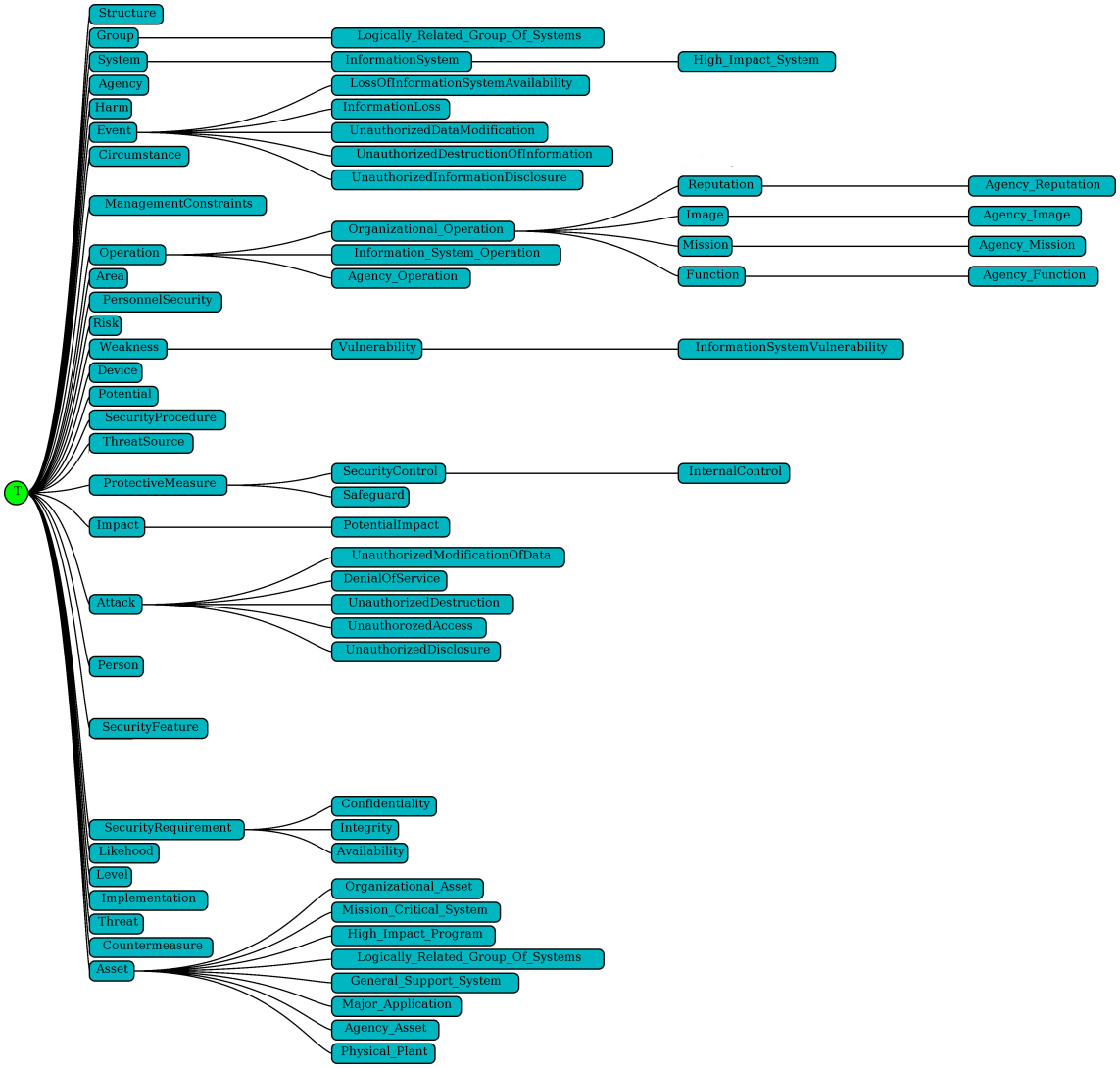


Rysunek 4.5: Ontologia bezpieczeństwa utworzona na bazie słownika ENISA – wywnioskowana hierarchia klas

cept *Organization* jest potomny względem konceptu *Group*. W związku z tym, algorytm musi zbadać jeszcze zależność klasy potomnej pojęcia *NIST:Organization*, jaką jest pojęcie *NIST:LogicallyRelatedGroupOfSystems* a pojęciem *ENISA:Organization*. W przypadku pojęć o wieloczłonowych nazwach w celu określenia podobieństwa stosowana jest miara podobieństwa identyczna jak w przypadku komentarzy. W tym przypadku nazwy są dzielone po białych znakach, oraz znakach specjalnych typu: ', ", [,], {, }, (,), :, ;, _ oraz -. Wynik obliczeń jest następujący: $P_{lex} = 0$, $P_{sem} = 0,25$, a $P_{sk} = 0,3$. P_{sk} wyznaczone zostało na podstawie równania 4.18, bazując na wartościach: $P_{str} = 1$ (oba pojęcia nie posiadają klas potomnych oraz dziedziczą po jednej klasie), $P_{kom} = 0$ (oba pojęcia nie posiadają komentarzy). W wynikowej ontologii utworzone zostanie więc pojęcie *Group*, a następnie jako potomne względem niego koncepty *Organization* oraz *NIST:LogicallyRelatedGroupOfSystems*.

Kolejnym analizowanym pojęciem jest *NIST:System*. W tym przypadku $P_{lex} = 1$. Wynika to z faktu, że w strukturze WordNet pojęcie *System* występuje jako synonim pojęcia *Organization*. Algorytm, w docelowej ontologii utworzy więc oba te pojęcia i powiąże je relacją tożsamości.

Porównując koncept *ENISA:Organization* z pojęciem *NIST:Agency* algorytm określi podobieństwo semantyczne $P_{sem} = 0,72$. Algorytm uznaje więc oba pojęcia za podobne. Ponadto w słowniku WordNet Synset opisujący pojęcie *NIST:Agency* jest hierarchicznie związany z pojęciem



Rysunek 4.6: Ontologia bezpieczeństwa utworzona na bazie słownika NIST – wywnioskowana hierarchia klas

ENISA:Organization. *NIST:Agency* należy do poddrzewa, którego korzeniem jest *ENISA:Organization*, a odległość pomiędzy węzłami wynosi 3. Oba węzły nie mają potomków w swoich ontologiach, więc algorytm utworzy w wynikowej ontologii węzeł reprezentujący pojęcia *Organization* oraz *Agency*, a następnie powiąże je aksjomatem potomności tak, że *Agency* będzie podklasą *Organization*.

Algorytm następnie przeprowadzi analogiczne porównanie konceptu *ENISA:Organization* z kolejnym konceptem z ontologii NIST, będącym bezpośrednimi potomkiem węzła *owl:Thing*, czyli *NIST:Harm*. Koncept *ENISA:Organization* opatrzony jest komentarzem „an organized body of people with a particular purpose, e.g. a business (http://www.askoxford.com/concise_oed/orga-nization?view=uk)”, nie posiada klas potomnych, ale wchodzi w relację rozłączności z 10 klasami. Koncept *NIST:Harm* nie posiada komentarza, klas potomnych ani nie jest związany innymi relacjami.

Miara podobieństwa leksykalnego w tym przypadku zwraca wartość $P_{lex} = 0$ - wyrazy będące etykietami konceptów są odmiennymi wyrazami nie będącymi synonimami. Podobieństwo semantyczne pomiędzy wyrazami wg algorytmu Lin wynosi $P_{sem} = 0.48$. Obie podstawowe miary

podobieństwa nie są więc w stanie rozstrzygnąć o zbieżności analizowanych pojęć. Algorytm bada więc podobieństwo strukturalne elementów oraz semantyczne ich komentarzy. Na podstawie równania 4.17 wartość podobieństwa strukturalnego $P_{str} = 0,0$. Podobieństwo komentarzy P_{kom}, wyrażone równaniem 4.16, również przyjmuje wartość $P_{kom} = 0,0$. Na podstawie tych wartości, z równania 4.18, otrzymujemy $P_{sk} = 0$. Wszystkie miary podobieństwa określają więc oba elementy jako rozłączne.

Podobnie algorytm zachowa się w kolejnym kroku. Koncept *ENISA:Organization* porównywany jest z pojęciem *NIST:Event*. Koncept *ENISA:Organization* opatrzony jest komentarzem „an organized body of people with a particular purpose, e.g. a business (http://www.askoxford.com/concise_oed/orga-nization?view=uk)", nie posiada klas potomnych oraz wchodzi w relację rozłączności z 10 klasami. Koncept *NIST:Event* opatrzony jest komentarzem „Any observable occurrence in a network or system. SOURCE: SP 800-61", posiada 5 klas potomnych oraz wchodzi w relację rozłączności z 2 klasami.

Miara podobieństwa leksykalnego w tym przypadku zwraca wartość $P_{lex} = 0$ - wyrazy będące etykietami konceptów są odmiennymi wyrazami nie będącymi synonimami. Podobieństwo semantyczne pomiędzy wyrazami wg algorytmu Lin wynosi $P_{sem} = 0,41$. Obie podstawowe miary podobieństwa nie są więc w stanie rozstrzygnąć o zbieżności analizowanych pojęć. Algorytm bada więc podobieństwo strukturalne elementów oraz semantyczne ich komentarzy. Na podstawie równania 4.17 wartość podobieństwa strukturalnego $P_{str} = 0,17$. Podobieństwo komentarzy P_{kom}, wyrażone równaniem 4.16, wynosi $0,22$. Na podstawie tych wartości, z równania 4.18, otrzymujemy $P_{sk} = 0,21$. Wszystkie miary podobieństwa określają oba elementy jako rozłączne.

Następnie algorytm wykona analogiczne porównania dla konceptów: *NIST:Circumstance*, *NIST:ManagementConstraintsNIST:Operation*, *NIST:Area*, *NIST:PersonnelSecurity*, *NIST:Risk*, *NIST:Weakness*,*NIST:Device*, *NIST:Potential*, *NIST:SecurityProcedure*, *NIST:ThreatSource*, *NIST:ProtectiveMeasure* oraz *NIST:Impact*. Ponownie każde z tych porównań wykaże rozłączność konceptów.

W przypadku porównania z pojęciem *NIST:Attack* algorytm zachowa się odmiennie. W wyniku obliczeń otrzymuje następujące rezultaty: $P_{lex} = 0$, $P_{sem} = 0,76$. Wyniki wskazują więc, że pojęcia są podobne. Jednakże analiza struktury słownika WordNet wskazuje, że oba porównywane pojęcia mają wspólnego rodzica *Event*. W związku z tym, w docelowej ontologii umieszczone powinny zostać oba elementy jako niezależne klasy, gdyż z punktu widzenia semantyki koncepty te są rodzeństwem.

Algorytm kontynuuje pracę analizując podobieństwo pojęcia *ENISA:Organization* do pozostałych potomków węzła *owl:Thing* ontologii NIST, czyli: *NIST:Person*, *NIST:SecurityFeature*, *NIST:SecurityRequirement*, *NIST:Likehood*, *NIST:Level*, *NIST:Implementation*, *NIST:Countermeasure*, *NIST:Threat* oraz *NIST:Asset*. Analogicznie jak w pierwszym przypadku, wynik działania algorytmu wskaże na rozłączność każdego z wymienionych pojęć z konceptem *ENISA:Organization*. Jako, że pojęcie to zostało już dodane do wynikowej ontologii algorytm przechodzi do porównania kolejnego potomka węzła *owl:Thing* ontologi ENISA z elementami ontologii NIST. Będzie to *ENISA:Mission*.

Podobnie jak w przypadku *ENISA:Organization* algorytm porównuje koncept *ENISA:Mission* kolejno z pojęciami potomnymi względem *owl:Thing* ontologii NIST. W przypadku *NIST:Structure*, *NIST:Group*, *NIST:System*, *NIST:Agency*, *NIST:Harm*, *NIST:Event*, *NIST:Circumstance*, *NIST:ManagementConstraints* otrzymujemy wynik wskazujący na różnorodność tych pojęć.

W przypadku *NIST:Operation* wynik porównania leksykalnego $P_{lex} = 0$, a wynik miary podobieństwa semantycznego $P_{sem} = 0,80$. Ponadto, jak już wcześniej zauważono, w strukturze słownika WordNet pojęcie *Mission* jest bezpośrednim potomkiem pojęcia *Operation*. W wynikowej ontologii umieszczony zostanie więc koncept *NIST:Operation*, a algorytm rozpocznie porównanie pojęcia *ENISA:Mission* z jego bezpośrednimi potomkami. Są to: *NIST:AgencyOperation*, *NIST:OrganizationalOperation* oraz *NIST:InformationSystemOperation*.

Porównania wyglądają następująco:

- *ENISA:Mission* i *NIST:AgencyOperation* – $P_{lex} = 0$, $P_{sem} = 0,40$. Oba pojęcia nie posiadają komentarzy, więc $P_{kom} = 0$. Oba pojęcia dziedziczą po jednej klasie (*owl:Thing*), pierwsze pojęcie ma jedną klasę potomną, drugie nie posiada potomków. Na podstawie równania 4.17 otrzymujemy $P_{str} = 0,50$. Na podstawie tych wartości, z równania 4.18, otrzymujemy $P_{sk} = 0,15$,
- *ENISA:Mission* i *NIST:OrganizationalOperation* – $P_{lex} = 0$, $P_{sem} = 0,40$. Oba pojęcia nie posiadają komentarzy, więc $P_{kom} = 0$. Oba pojęcia dziedziczą po jednej klasie (*owl:Thing*), pierwsze pojęcie ma jedną klasą potomną, drugie posiada czterech potomków. Na podstawie równania 4.17 otrzymujemy $P_{str} = 0,40$. Na podstawie tych wartości, z równania 4.18, otrzymujemy $P_{sk} = 0,12$,
- *ENISA:Mission* i *NIST:InformationSystemOperation* – $P_{lex} = 0$, $P_{sem} = 0,27$. Oba pojęcia nie posiadają komentarzy, więc $P_{kom} = 0$. Oba pojęcia dziedziczą po jednej klasie (*owl:Thing*), pierwsze pojęcie ma jedną klasę potomną, drugie nie posiada potomków. Na podstawie równania 4.17 otrzymujemy $P_{str} = 0,50$. Na podstawie tych wartości, z równania 4.18, otrzymujemy $P_{sk} = 0,15$.

Na podstawie wymienionych wartości podobieństwa algorytm określa, że poszczególne pary pojęć są rozłączne. W wynikowej ontologii utworzony zostanie więc koncept *Mission*, jako podrzędny względem *Operation*. Będąc rozłącznymi z konceptami podrzędnymi względem pojęcia *Operation*, koncepty te zostaną zdefiniowane jako sąsiedzi pojęcia *Mission*.

Po zakończeniu analizy poddrzew łączonych konceptów algorytm kontynuuje pracę porównując pojęcie *ENISA:Mission* z pozostałymi potomkami *owl:Thing* ontologii NIST. Są to: *NIST:Area*, *NIST:PersonnelSecurity*, *NIST:Risk*, *NIST:Weakness*, *NIST:Device*, *NIST:Potential*, *NIST:SecurityProcedure*, *NIST:ThreatSource*, *NIST:ProtectiveMeasure*, *NIST:Impact*. Wynik działania algorytmu wskaże na rozłączność każdego z wymienionych pojęć z pojęciem *ENISA:Mission*.

W przypadku porównania z *NIST:Attack* uzyskujemy podobieństwo semantyczne $P_{sem} = 0,77$. Wskazuje to na duże podobieństwo obu pojęć. Analiza struktury słownika WordNet wskazuje jednak, że oba pojęcia mają jednego wspólnego przodka, jakim jest pojęcie *Operation*. W związku z tym, w docelowej ontologii umieszczone powinny zostać oba elementy jako niezależne klasy, gdyż z punktu widzenia semantyki koncepty *ENISA:Mission* oraz *NIST:Attack* są rodzeństwem. Ponadto zauważmy, że węzeł będący rodzicem obu pojęć występuje w ontologii NIST. Nie chcąc jednak wpływać na oryginalną strukturę ontologii, klasa *NIST:Attack* nie zostanie zmodyfikowana w celu dziedziczenia po *NIST:Operation*.

Algorytm kontynuuje swoją pracę dla pozostałych pojęć w ontologii NIST, czyli: *NIST:Person*, *NIST:SecurityFeature*, *NIST:SecurityRequirement*, *NIST:Likehood*, *NIST:Level*, *NIST:Implementation*, *NIST:Threat*, *NIST:Countermeasure* oraz *NIST:Asset*. Każde z tych pojęć jest rozłączne z *ENISA:Mission*.

W analogiczny sposób algorytm kontynuuje pracę porównując pozostałe pojęcia z ontologii *ENISA* z pojęciami z ontologii *NIST*. Wyniki poszczególnych porównań przedstawiono w tablicach 4.3 oraz 4.4. Wyniki te zostały rozbite na poszczególne tabele w celu poprawy czytelności. Tablica 4.3 prezentuje wzajemne porównania pojęć głównych (dziedziczących bezpośrednio po *owl:Thing*) obu integrowanych ontologii. Tablice 4.4 oraz 4.5 prezentują wyniki dodatkowych porównań wykonanych w poddrzewach pojęć określonych w wyniku pierwszego porównania jako podrzędne lub rodzeństwo. We wszystkich wypadkach pojęcia oznaczone jako „Lemma A" pochodzą z ontologii ENISA a „Lemma B" z ontologii NIST.

Tablica 4.3: Wyniki wzajemnych porównań pojęć głównych (dziedziczących bezpośrednio po *owl:Thing*) obu integrowanych ontologii

Lemma B	Agency					Area				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,30	0,00	0,06	0,02	Różne	0,40	0,00	0,06	0,02	Różne
Attack	0,33	0,00	0,09	0,03	Różne	0,33	0,00	0,09	0,03	Różne
Circumstance	0,37	0,00	0,00	0,00	Różne	0,85	X	X	X	Rodzeństwo
Continuity	0,31	0,00	0,13	0,04	Różne	0,42	0,00	0,13	0,04	Różne
Error	0,32	0,00	0,08	0,03	Różne	0,35	0,00	0,08	0,03	Różne
Event	0,46	0,00	0,10	0,03	Różne	0,40	0,00	0,10	0,03	Różne
Group	0,60	0,00	0,00	0,00	Różne	0,55	0,00	0,00	0,00	Różne
Harm	0,37	0,00	0,00	0,00	Różne	0,42	0,00	0,00	0,00	Różne
Mission	0,46	0,00	0,10	0,03	Różne	0,19	0,00	0,10	0,03	Różne
Operation	0,83	X	X	X	Rodzeństwo	0,43	0,00	0,50	0,15	Różne
Organization	0,72	X	X	X	B podrzędne A	0,35	0,00	0,10	0,03	Różne
Potential	0,36	0,00	0,00	0,00	Różne	0,41	0,00	0,00	0,00	Różne
Risk	0,32	0,00	0,17	0,05	Różne	0,20	0,00	0,17	0,05	Różne
Security	0,65	0,00	0,00	0,00	Różne	0,40	0,00	0,00	0,00	Różne
Threat	0,28	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Value	0,30	0,00	0,11	0,03	Różne	0,47	0,00	0,11	0,03	Różne
Vulnerability	0,31	0,00	0,00	0,00	Różne	0,42	0,00	0,00	0,00	Różne
Weakness	0,34	0,00	0,11	0,03	Różne	0,44	0,00	0,11	0,03	Różne

Lemma B	Asset					Attack				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	1,00	X	X	X	Tożsame	0,17	0,00	0,39	0,12	Różne
Attack	0,17	0,00	0,85	0,25	Różne	1,00	X	X	X	Tożsame
Circumstance	0,32	0,34	0,00	0,24	Różne	0,25	0,00	0,00	0,00	Różne
Continuity	0,27	0,00	0,50	0,15	Różne	0,26	0,00	0,25	0,08	Różne
Error	0,48	0,25	0,47	0,31	Różne	0,45	0,00	0,27	0,08	Różne
Event	0,32	0,20	0,64	0,33	Różne	0,51	0,00	0,42	0,13	Różne
Group	0,13	0,00	0,00	0,00	Różne	0,31	0,00	0,00	0,00	Różne
Harm	0,33	0,26	0,00	0,18	Różne	0,51	0,00	0,00	0,00	Różne
Mission	0,29	0,00	0,44	0,13	Różne	0,77	X	X	X	Rodzeństwo
Operation	0,31	0,00	0,15	0,05	Różne	0,95	X	X	X	B podrzędne A
Organization	0,55	0,31	0,35	0,33	Różne	0,76	X	X	X	Rodzeństwo
Potential	0,32	0,22	0,00	0,16	Różne	0,46	0,00	0,00	0,00	Różne
Risk	0,20	0,29	0,46	0,34	Różne	0,40	0,00	0,18	0,05	Różne
Security	0,31	0,00	0,00	0,00	Różne	0,39	0,00	0,00	0,00	Różne
Threat	0,33	0,23	0,00	0,16	Różne	0,40	0,00	0,00	0,00	Różne
Value	0,64	0,00	0,38	0,11	Różne	0,20	0,00	0,14	0,04	Różne
Vulnerability	0,27	0,31	0,00	0,22	Różne	0,08	0,00	0,00	0,00	Różne
Weakness	0,47	0,22	0,38	0,27	Różne	0,13	0,00	0,14	0,04	Różne

Lemma B	Circumstance					Countermeasure				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,32	0,00	0,11	0,03	Różne	0,14	0,00	0,00	0,00	Różne
Attack	0,25	0,00	0,18	0,05	Różne	0,31	0,00	0,00	0,00	Różne
Circumstance	1,00	X	X	X	Tożsame	0,21	0,00	0,00	0,00	Różne
Continuity	0,33	0,00	0,25	0,08	Różne	0,29	0,00	0,00	0,00	Różne
Error	0,27	0,00	0,17	0,05	Różne	0,21	0,00	0,00	0,00	Różne
Event	0,99	X	X	X	A podrzędne B	0,26	0,00	0,00	0,00	Różne
Group	0,17	0,00	0,00	0,00	Różne	0,14	0,00	0,00	0,00	Różne
Harm	0,52	0,00	0,00	0,00	Różne	0,33	0,00	0,00	0,00	Różne
Mission	0,25	0,00	0,20	0,06	Różne	0,20	0,00	0,00	0,00	Różne
Operation	0,41	0,00	0,33	0,10	Różne	0,29	0,00	0,00	0,00	Różne
Organization	0,34	0,00	0,20	0,06	Różne	0,34	0,00	0,00	0,00	Różne
Potential	0,39	0,00	0,00	0,00	Różne	0,29	0,00	0,00	0,00	Różne
Risk	0,23	0,00	0,33	0,10	Różne	0,21	0,00	0,00	0,00	Różne
Security	0,49	0,00	0,00	0,00	Różne	0,73	X	X	X	Rodzeństwo
Threat	0,21	0,00	0,00	0,00	Różne	0,29	0,00	0,00	0,00	Różne

Tablica 4.3: (ciąg dalszy)

Value	0,33	0,00	0,22	0,07	Różne	0,21	0,00	0,00	0,00	Różne
Vulnerability	0,43	0,00	0,00	0,00	Różne	0,21	0,00	0,00	0,00	Różne
Weakness	0,36	0,00	0,22	0,07	Różne	0,14	0,00	0,00	0,00	Różne

Lemma B	Device					Event				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,08	0,00	0,00	0,00	Różne	0,32	0,13	0,39	0,21	Różne
Attack	0,37	0,00	0,00	0,00	Różne	0,51	0,00	0,64	0,19	Różne
Circumstance	0,21	0,00	0,00	0,00	Różne	0,99	X	X	X	B podrzędne A
Continuity	0,27	0,00	0,00	0,00	Różne	0,32	0,00	0,25	0,08	Różne
Error	0,30	0,00	0,00	0,00	Różne	0,42	0,17	0,27	0,20	Różne
Event	0,34	0,00	0,00	0,00	Różne	1,00	X	X	X	Tożsame
Group	0,13	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Harm	0,40	0,00	0,00	0,00	Różne	0,51	0,18	0,00	0,12	Różne
Mission	0,24	0,00	0,00	0,00	Różne	0,37	0,00	0,21	0,06	Różne
Operation	0,27	0,00	0,00	0,00	Różne	0,57	0,00	0,29	0,09	Różne
Organization	0,41	0,00	0,00	0,00	Różne	0,41	0,22	0,13	0,20	Różne
Potential	0,22	0,00	0,00	0,00	Różne	0,67	0,14	0,00	0,10	Różne
Risk	0,26	0,00	0,00	0,00	Różne	0,41	0,18	0,18	0,18	Różne
Security	0,83	X	X	X	Rodzeństwo	0,48	0,00	0,00	0,00	Różne
Threat	0,30	0,00	0,00	0,00	Różne	0,35	0,19	0,00	0,13	Różne
Value	0,18	0,00	0,00	0,00	Różne	0,32	0,00	0,14	0,04	Różne
Vulnerability	0,15	0,00	0,00	0,00	Różne	0,42	0,18	0,00	0,12	Różne
Weakness	0,25	0,00	0,00	0,00	Różne	0,35	0,23	0,14	0,21	Różne

Lemma B	Group					Harm				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,13	0,00	0,00	0,00	Różne	0,33	0,00	0,00	0,00	Różne
Attack	0,31	0,00	0,00	0,00	Różne	0,51	0,00	0,00	0,00	Różne
Circumstance	0,17	0,00	0,00	0,00	Różne	0,52	0,00	0,00	0,00	Różne
Continuity	0,10	0,00	0,00	0,00	Różne	0,33	0,00	0,00	0,00	Różne
Error	0,20	0,00	0,00	0,00	Różne	0,45	0,00	0,00	0,00	Różne
Event	0,24	0,00	0,00	0,00	Różne	0,51	0,00	0,00	0,00	Różne
Group	1,00	X	X	X	Tożsame	0,11	0,00	0,00	0,00	Różne
Harm	0,11	0,00	0,00	0,00	Różne	1,00	X	X	X	Tożsame
Mission	0,46	0,00	0,00	0,00	Różne	0,26	0,00	0,00	0,00	Różne
Operation	0,17	0,00	0,00	0,00	Różne	0,39	0,00	0,00	0,00	Różne
Organization	0,87	X	X	X	A podrzędne B	0,48	0,00	0,00	0,00	Różne
Potential	0,11	0,00	0,00	0,00	Różne	0,40	0,00	0,00	0,00	Różne
Risk	0,32	0,00	0,00	0,00	Różne	0,28	0,00	0,00	0,00	Różne
Security	0,43	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Threat	0,17	0,00	0,00	0,00	Różne	0,25	0,00	0,00	0,00	Różne
Value	0,46	0,00	0,00	0,00	Różne	0,33	0,00	0,00	0,00	Różne
Vulnerability	0,09	0,00	0,00	0,00	Różne	0,44	0,00	0,00	0,00	Różne
Weakness	0,10	0,00	0,00	0,00	Różne	0,36	0,00	0,00	0,00	Różne

Lemma B	Impact					Implementation				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,17	0,35	0,11	0,28	Różne	0,14	0,00	0,06	0,02	Różne
Attack	0,50	0,00	0,08	0,03	Różne	0,32	0,00	0,09	0,03	Różne
Circumstance	0,33	0,32	0,00	0,22	Różne	0,25	0,00	0,00	0,00	Różne
Continuity	0,10	0,00	0,11	0,03	Różne	0,14	0,00	0,13	0,04	Różne
Error	0,45	0,31	0,08	0,24	Różne	0,32	0,00	0,08	0,03	Różne
Event	0,81	X	X	X	B podrzędne A	0,47	0,00	0,10	0,03	Różne
Group	0,10	0,00	0,00	0,00	Różne	0,12	0,00	0,00	0,00	Różne
Harm	0,50	0,27	0,00	0,19	Różne	0,30	0,00	0,00	0,00	Różne
Mission	0,25	0,00	0,09	0,03	Różne	0,29	0,00	0,10	0,03	Różne
Operation	0,48	0,00	0,33	0,10	Różne	0,50	0,00	0,50	0,15	Różne
Organization	0,47	0,28	0,00	0,19	Różne	0,61	0,00	0,10	0,03	Różne
Potential	0,55	0,31	0,00	0,22	Różne	0,36	0,00	0,00	0,00	Różne
Risk	0,27	0,33	0,00	0,23	Różne	0,32	0,00	0,17	0,05	Różne

Tablica 4.3: (ciąg dalszy)

Security	0,24	0,00	0,00	0,00	Różne	0,28	0,00	0,00	0,00	Różne
Threat	0,24	0,46	0,33	0,42	Różne	0,28	0,00	0,00	0,00	Różne
Value	0,20	0,00	0,00	0,00	Różne	0,22	0,00	0,11	0,03	Różne
Vulnerability	0,15	0,42	0,33	0,39	Różne	0,14	0,00	0,00	0,00	Różne
Weakness	0,07	0,27	0,00	0,19	Różne	0,14	0,00	0,11	0,03	Różne

Lemma B	Level					Likehood				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,35	0,00	0,11	0,03	Różne	0,13	0,00	0,06	0,02	Różne
Attack	0,27	0,00	0,18	0,05	Różne	0,00	0,00	0,09	0,03	Różne
Circumstance	0,41	0,00	0,00	0,00	Różne	0,08	0,00	0,00	0,00	Różne
Continuity	0,44	0,00	0,25	0,08	Różne	0,00	0,00	0,13	0,04	Różne
Error	0,27	0,00	0,17	0,05	Różne	0,25	0,00	0,08	0,03	Różne
Event	0,40	0,00	0,20	0,06	Różne	0,13	0,00	0,10	0,03	Różne
Group	0,33	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne
Harm	0,42	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne
Mission	0,18	0,00	0,20	0,06	Różne	0,25	0,00	0,10	0,03	Różne
Operation	0,41	0,00	0,33	0,10	Różne	0,11	0,00	0,50	0,15	Różne
Organization	0,34	0,00	0,20	0,06	Różne	0,17	0,00	0,10	0,03	Różne
Potential	0,41	0,00	0,00	0,00	Różne	0,11	0,00	0,00	0,00	Różne
Risk	0,28	0,00	0,33	0,10	Różne	0,13	0,00	0,17	0,05	Różne
Security	0,45	0,00	0,00	0,00	Różne	0,00	0,00	0,00	0,00	Różne
Threat	0,18	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne
Value	0,49	0,00	0,22	0,07	Różne	0,00	0,00	0,11	0,03	Różne
Vulnerability	0,44	0,00	0,00	0,00	Różne	0,08	0,00	0,00	0,00	Różne
Weakness	0,46	0,00	0,22	0,07	Różne	0,00	0,00	0,11	0,03	Różne

Lemma B	ManagementConstraints					Operation				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,15	0,00	0,00	0,00	Różne	0,31	0,00	0,17	0,05	Różne
Attack	0,17	0,00	0,00	0,00	Różne	0,95	X	X	X	A podrzędne B
Circumstance	0,17	0,00	0,00	0,00	Różne	0,41	0,00	0,00	0,00	Różne
Continuity	0,23	0,00	0,00	0,00	Różne	0,32	0,00	0,10	0,03	Różne
Error	0,16	0,00	0,00	0,00	Różne	0,36	0,00	0,15	0,05	Różne
Event	0,24	0,00	0,00	0,00	Różne	0,57	0,00	0,30	0,09	Różne
Group	0,26	0,00	0,00	0,00	Różne	0,17	0,00	0,00	0,00	Różne
Harm	0,18	0,00	0,00	0,00	Różne	0,39	0,00	0,00	0,00	Różne
Mission	0,19	0,00	0,00	0,00	Różne	0,80	X	X	X	A podrzędne B
Operation	0,18	0,00	0,00	0,00	Różne	1,00	X	X	X	Tożsame
Organization	0,48	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Potential	0,17	0,00	0,00	0,00	Różne	0,51	0,00	0,00	0,00	Różne
Risk	0,17	0,00	0,00	0,00	Różne	0,68	0,00	0,00	0,00	Różne
Security	0,24	0,00	0,00	0,00	Różne	0,37	0,00	0,00	0,00	Różne
Threat	0,14	0,00	0,00	0,00	Różne	0,28	0,00	0,00	0,00	Różne
Value	0,14	0,00	0,00	0,00	Różne	0,43	0,00	0,00	0,00	Różne
Vulnerability	0,15	0,00	0,00	0,00	Różne	0,32	0,00	0,00	0,00	Różne
Weakness	0,16	0,00	0,00	0,00	Różne	0,35	0,00	0,00	0,00	Różne

Lemma B	Person					PersonnelSecurity				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,17	0,00	0,00	0,00	Różne	0,16	0,00	0,00	0,00	Różne
Attack	0,12	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Circumstance	0,25	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Continuity	0,10	0,00	0,00	0,00	Różne	0,22	0,00	0,00	0,00	Różne
Error	0,50	0,00	0,08	0,03	Różne	0,20	0,00	0,00	0,00	Różne
Event	0,17	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Group	0,50	0,00	1,00	0,30	Różne	0,31	0,00	0,00	0,00	Różne
Harm	0,17	0,00	1,00	0,30	Różne	0,25	0,00	0,00	0,00	Różne
Mission	0,43	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Operation	0,56	0,00	0,00	0,00	Różne	0,18	0,00	0,00	0,00	Różne
Organization	0,47	0,00	0,00	0,00	Różne	0,37	0,00	0,00	0,00	Różne

Tablica 4.3: (ciąg dalszy)

Potential	0,22	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Risk	0,39	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne
Security	0,30	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Threat	0,37	0,00	0,50	0,15	Różne	0,19	0,00	0,00	0,00	Różne
Value	0,08	0,00	0,00	0,00	Różne	0,16	0,00	0,00	0,00	Różne
Vulnerability	0,15	0,00	0,50	0,15	Różne	0,21	0,00	0,00	0,00	Różne
Weakness	0,13	0,00	0,00	0,00	Różne	0,25	0,00	0,00	0,00	Różne

Lemma B	Potential					ProtectiveMeasure				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,32	0,00	0,00	0,00	Różne	0,14	0,00	0,11	0,03	Różne
Attack	0,46	0,00	0,00	0,00	Różne	0,20	0,00	0,18	0,05	Różne
Circumstance	0,39	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne
Continuity	0,40	0,00	0,00	0,00	Różne	0,21	0,00	0,11	0,03	Różne
Error	0,25	0,00	0,00	0,00	Różne	0,15	0,00	0,17	0,05	Różne
Event	0,67	0,00	0,00	0,00	Różne	0,21	0,00	0,20	0,06	Różne
Group	0,11	0,00	0,00	0,00	Różne	0,23	0,00	0,00	0,00	Różne
Harm	0,40	0,00	0,00	0,00	Różne	0,21	0,00	0,00	0,00	Różne
Mission	0,11	0,00	0,00	0,00	Różne	0,18	0,00	0,09	0,03	Różne
Operation	0,51	0,00	0,00	0,00	Różne	0,21	0,00	0,33	0,10	Różne
Organization	0,28	0,00	0,00	0,00	Różne	0,22	0,00	0,00	0,00	Różne
Potential	1,00	X	X	X	Tożsame	0,30	0,00	0,00	0,00	Różne
Risk	0,11	0,00	0,00	0,00	Różne	0,21	0,00	0,00	0,00	Różne
Security	0,38	0,00	0,00	0,00	Różne	0,47	0,00	0,00	0,00	Różne
Threat	0,22	0,00	0,00	0,00	Różne	0,14	0,00	0,00	0,00	Różne
Value	0,32	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Vulnerability	0,33	0,00	0,00	0,00	Różne	0,08	0,00	0,00	0,00	Różne
Weakness	0,36	0,00	0,00	0,00	Różne	0,13	0,00	0,00	0,00	Różne

Lemma B	Risk					SecurityFeature				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,20	0,47	0,06	0,35	Różne	0,16	0,00	0,00	0,00	Różne
Attack	0,40	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Circumstance	0,23	0,23	0,00	0,16	Różne	0,27	0,00	0,00	0,00	Różne
Continuity	0,10	0,00	0,00	0,00	Różne	0,22	0,00	0,00	0,00	Różne
Error	0,30	0,28	0,00	0,19	Różne	0,22	0,00	0,00	0,00	Różne
Event	0,41	0,30	0,00	0,21	Różne	0,24	0,00	0,00	0,00	Różne
Group	0,32	0,00	0,00	0,00	Różne	0,25	0,00	0,00	0,00	Różne
Harm	0,28	0,19	0,00	0,13	Różne	0,25	0,00	0,00	0,00	Różne
Mission	0,56	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Operation	0,68	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Organization	0,39	0,24	0,00	0,17	Różne	0,26	0,00	0,00	0,00	Różne
Potential	0,11	0,31	0,00	0,22	Różne	0,19	0,00	0,00	0,00	Różne
Risk	1,00	X	X	X	Tożsame	0,13	0,00	0,00	0,00	Różne
Security	0,26	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Threat	0,89	X	X	X	Rodzeństwo	0,19	0,00	0,00	0,00	Różne
Value	0,26	0,00	0,00	0,00	Różne	0,28	0,00	0,00	0,00	Różne
Vulnerability	0,15	0,40	0,50	0,43	Różne	0,21	0,00	0,00	0,00	Różne
Weakness	0,13	0,23	0,00	0,16	Różne	0,25	0,00	0,00	0,00	Różne

Lemma B	SecurityProcedure					SecurityRequirement				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,16	0,00	0,00	0,00	Różne	0,16	0,00	0,17	0,05	Różne
Attack	0,22	0,00	0,00	0,00	Różne	0,19	0,00	0,27	0,08	Różne
Circumstance	0,24	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Continuity	0,22	0,00	0,00	0,00	Różne	0,22	0,00	0,10	0,03	Różne
Error	0,20	0,00	0,00	0,00	Różne	0,20	0,00	0,15	0,05	Różne
Event	0,24	0,00	0,00	0,00	Różne	0,24	0,00	0,30	0,09	Różne
Group	0,21	0,00	0,00	0,00	Różne	0,30	0,00	0,00	0,00	Różne
Harm	0,25	0,00	0,00	0,00	Różne	0,25	0,00	0,00	0,00	Różne
Mission	0,26	0,00	0,00	0,00	Różne	0,19	0,00	0,08	0,03	Różne

Tablica 4.3: (ciąg dalszy)

Operation	0,50	0,00	0,00	0,00	Różne	0,18	0,00	0,25	0,08	Różne
Organization	0,26	0,00	0,00	0,00	Różne	0,32	0,00	0,00	0,00	Różne
Potential	0,19	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Risk	0,28	0,00	0,00	0,00	Różne	0,14	0,00	0,00	0,00	Różne
Security	0,50	0,00	0,00	0,00	Różne	0,50	0,00	0,00	0,00	Różne
Threat	0,19	0,00	0,00	0,00	Różne	0,19	0,00	0,00	0,00	Różne
Value	0,16	0,00	0,00	0,00	Różne	0,16	0,00	0,00	0,00	Różne
Vulnerability	0,21	0,00	0,00	0,00	Różne	0,21	0,00	0,00	0,00	Różne
Weakness	0,25	0,00	0,00	0,00	Różne	0,25	0,00	0,00	0,00	Różne

Lemma B	Structure					System				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,32	0,00	0,00	0,00	Różne	0,55	0,31	0,06	0,23	Różne
Attack	0,23	0,00	0,00	0,00	Różne	0,22	0,00	0,09	0,03	Różne
Circumstance	0,34	0,00	0,00	0,00	Różne	0,34	0,24	0,00	0,17	Różne
Continuity	0,41	0,00	0,00	0,00	Różne	0,24	0,00	0,13	0,04	Różne
Error	0,31	0,00	0,00	0,00	Różne	0,43	0,27	0,08	0,21	Różne
Event	0,37	0,00	0,00	0,00	Różne	0,36	0,32	0,10	0,25	Różne
Group	0,59	0,00	0,00	0,00	Różne	0,59	0,00	0,00	0,00	Różne
Harm	0,29	0,00	0,00	0,00	Różne	0,29	0,28	0,00	0,20	Różne
Mission	0,31	0,00	0,00	0,00	Różne	0,35	0,00	0,10	0,03	Różne
Operation	0,49	0,00	0,00	0,00	Różne	0,48	0,00	0,50	0,15	Różne
Organization	0,98	X	X	X	A podrzędne B	1,00	X	X	X	Tożsame
Potential	0,28	0,00	0,00	0,00	Różne	0,28	0,21	0,00	0,15	Różne
Risk	0,22	0,00	0,00	0,00	Różne	0,22	0,36	0,00	0,25	Różne
Security	0,34	0,00	0,00	0,00	Różne	0,39	0,00	0,00	0,00	Różne
Threat	0,22	0,00	0,00	0,00	Różne	0,20	0,31	0,00	0,22	Różne
Value	0,45	0,00	0,00	0,00	Różne	0,55	0,00	0,00	0,00	Różne
Vulnerability	0,41	0,00	0,00	0,00	Różne	0,24	0,26	0,00	0,18	Różne
Weakness	0,43	0,00	0,00	0,00	Różne	0,42	0,31	0,00	0,22	Różne

Lemma B	Threat					ThreatSource				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Asset	0,33	0,16	0,11	0,14	Różne	0,17	0,00	0,00	0,00	Różne
Attack	0,40	0,00	0,08	0,03	Różne	0,22	0,00	0,00	0,00	Różne
Circumstance	0,21	0,11	0,00	0,08	Różne	0,15	0,00	0,00	0,00	Różne
Continuity	0,26	0,00	0,11	0,03	Różne	0,23	0,00	0,00	0,00	Różne
Error	0,50	0,13	0,08	0,11	Różne	0,25	0,00	0,00	0,00	Różne
Event	0,35	0,17	0,09	0,15	Różne	0,27	0,00	0,00	0,00	Różne
Group	0,17	0,00	0,00	0,00	Różne	0,24	0,00	0,00	0,00	Różne
Harm	0,25	0,07	0,00	0,05	Różne	0,13	0,00	0,00	0,00	Różne
Mission	0,25	0,00	0,09	0,03	Różne	0,12	0,00	0,00	0,00	Różne
Operation	0,28	0,00	0,33	0,10	Różne	0,24	0,00	0,00	0,00	Różne
Organization	0,26	0,12	0,09	0,11	Różne	0,16	0,00	0,00	0,00	Różne
Potential	0,22	0,12	0,00	0,09	Różne	0,24	0,00	0,00	0,00	Różne
Risk	0,89	X	X	X	Rodzeństwo	0,45	0,00	0,00	0,00	Różne
Security	0,38	0,00	0,00	0,00	Różne	0,31	0,00	0,00	0,00	Różne
Threat	1,00	X	X	X	Tożsame	0,50	0,00	0,00	0,00	Różne
Value	0,18	0,00	0,10	0,03	Różne	0,22	0,00	0,00	0,00	Różne
Vulnerability	0,23	0,18	0,33	0,22	Różne	0,12	0,00	0,00	0,00	Różne
Weakness	0,13	0,10	0,10	0,10	Różne	0,06	0,00	0,00	0,00	Różne

Lemma B	Weakness									
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik					
Asset	0,47	0,00	0,44	0,13	Różne					
Attack	0,13	0,00	0,58	0,18	Różne					
Circumstance	0,36	0,00	0,00	0,00	Różne					
Continuity	0,37	0,00	1,00	0,30	Różne					
Error	0,38	0,00	0,67	0,20	Różne					
Event	0,35	0,00	0,80	0,24	Różne					
Group	0,10	0,00	0,00	0,00	Różne					

Tablica 4.3: (ciąg dalszy)

Harm	0,36	0,00	0,00	0,00	Różne					
Mission	0,06	0,00	0,80	0,24	Różne					
Operation	0,35	0,00	0,25	0,08	Różne					
Organization	0,42	0,00	0,64	0,19	Różne					
Potential	0,36	0,00	0,00	0,00	Różne					
Risk	0,13	0,00	0,75	0,23	Różne					
Security	0,50	0,00	0,00	0,00	Różne					
Threat	0,13	0,00	0,00	0,00	Różne					
Value	0,47	0,00	0,70	0,21	Różne					
Vulnerability	0,96	X	X	X	A podrzędne B					
Weakness	1,00	X	X	X	Tożsame					

Tablica 4.4: Wyniki wzajemnych porównań pojęć dziedziczących po konceptach innych niż *owl:Thing*) obu integrowanych ontologii

Lemma B	AgencyAsset					GeneralSupportSystem				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Application	0,16	0,00	0,00	0,00	Różne	0,13	0,00	1,00	0,30	Różne
Computersystem	0,21	0,00	0,00	0,00	Różne	0,54	0,00	1,00	0,30	Różne
Control	0,28	0,00	1,00	0,30	Różne	0,31	0,00	0,00	0,00	Różne
InformationResource	0,66	0,00	0,00	0,00	Różne	0,57	0,00	1,00	0,30	Różne
Mechanism	0,14	0,00	0,00	0,00	Różne	0,23	0,00	1,00	0,30	Różne
Network	0,17	0,00	0,00	0,00	Różne	0,31	0,00	1,00	0,30	Różne
Practice	0,42	0,00	0,00	0,00	Różne	0,15	0,00	1,00	0,30	Różne
Procedure	0,17	0,00	0,00	0,00	Różne	0,14	0,00	1,00	0,30	Różne
Protocol	0,09	0,00	0,00	0,00	Różne	0,10	0,00	1,00	0,30	Różne

Lemma B	HighImpactProgram					LogicallyRelatedGroupOfSystems				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Application	0,32	0,00	1,00	0,30	Różne	0,08	0,00	0,50	0,15	Różne
ComputerSystem	0,47	0,00	1,00	0,30	Różne	0,13	0,00	0,50	0,15	Różne
Control	0,24	0,00	0,00	0,00	Różne	0,12	0,00	0,50	0,15	Różne
InformationResource	0,36	0,00	1,00	0,30	Różne	0,29	0,00	0,50	0,15	Różne
Mechanism	0,27	0,00	1,00	0,30	Różne	0,14	0,00	0,50	0,15	Różne
Network	0,13	0,00	1,00	0,30	Różne	0,23	0,00	0,50	0,15	Różne
Practice	0,14	0,00	1,00	0,30	Różne	0,09	0,00	0,50	0,15	Różne
Procedure	0,28	0,00	1,00	0,30	Różne	0,08	0,00	0,50	0,15	Różne
Protocol	0,14	0,00	1,00	0,30	Różne	0,07	0,00	0,50	0,15	Różne

Lemma B	MajorApplication					MissionCriticalSystem				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Application	0,50	0,00	1,00	0,30	Różne	0,18	0,00	1,00	0,30	Różne
ComputerSystem	0,28	0,00	1,00	0,30	Różne	0,08	0,00	1,00	0,30	Różne
Control	0,22	0,00	0,00	0,00	Różne	0,16	0,00	0,00	0,00	Różne
InformationResource	0,40	0,00	1,00	0,30	Różne	0,17	0,00	1,00	0,30	Różne
Mechanism	0,19	0,00	1,00	0,30	Różne	0,19	0,00	1,00	0,30	Różne
Network	0,14	0,00	1,00	0,30	Różne	0,31	0,00	1,00	0,30	Różne
Practice	0,40	0,00	1,00	0,30	Różne	0,13	0,00	1,00	0,30	Różne
Procedure	0,40	0,00	1,00	0,30	Różne	0,18	0,00	1,00	0,30	Różne
Protocol	0,19	0,00	1,00	0,30	Różne	0,17	0,00	1,00	0,30	Różne

Lemma B	OrganizationalAsset					PhysicalPlant				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Application	0,21	0,00	1,00	0,30	Różne	0,14	0,00	1,00	0,30	Różne
ComputerSystem	0,07	0,00	1,00	0,30	Różne	0,13	0,00	1,00	0,30	Różne
Control	0,28	0,00	0,00	0,00	Różne	0,21	0,00	0,00	0,00	Różne
InformationResource	0,64	0,00	1,00	0,30	Różne	0,05	0,00	1,00	0,30	Różne
Mechanism	0,07	0,00	1,00	0,30	Różne	0,19	0,00	1,00	0,30	Różne
Network	0,11	0,00	1,00	0,30	Różne	0,16	0,00	1,00	0,30	Różne
Practice	0,14	0,00	1,00	0,30	Różne	0,19	0,00	1,00	0,30	Różne

Tablica 4.4: (ciąg dalszy)

Procedure	0,06	0,00	1,00	0,30	Różne	0,10	0,00	1,00	0,30	Różne
Protocol	0,14	0,00	1,00	0,30	Różne	0,19	0,00	1,00	0,30	Różne

Lemma B	AgencyOperation					InformationSystemOperation				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Attack	0,48	0,00	0,00	0,00	Różne	0,32	0,00	0,00	0,00	Różne
BusinessOperation	0,82	X	X	X	Tożsame	0,43	0,00	0,50	0,15	Różne
Mission	0,40	0,00	0,00	0,00	Różne	0,27	0,00	0,00	0,00	Różne

Lemma B	OrganizationalOperation				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Attack	0,48	0,00	0,33	0,10	Różne
BusinessOperation	0,66	0,00	0,40	0,12	Różne
Mission	0,40	0,00	0,07	0,02	Różne

Lemma B	DenialOfService					UnauthorizedDestruction				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
DenialOfServices	1,00	X	X	X	Tożsame	0,25	0,00	1,00	0,30	Różne
Destruction	0,31	0,00	1,00	0,30	Różne	0,50	0,00	1,00	0,30	Różne
Disclosure	0,34	0,00	1,00	0,30	Różne	0,14	0,00	1,00	0,30	Różne
ModificationOfData	0,42	0,00	1,00	0,30	Różne	0,36	0,00	1,00	0,30	Różne
UnauthorizedAccess	0,24	0,00	1,00	0,30	Różne	0,72	X	X	X	Tożsame

Lemma B	UnauthorizedDisclosure					UnauthorizedModificationOfData				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
DenialOfServices	0,43	0,00	1,00	0,30	Różne	0,13	0,00	1,00	0,30	Różne
Destruction	0,14	0,00	1,00	0,30	Różne	0,18	0,00	1,00	0,30	Różne
Disclosure	0,50	0,00	1,00	0,30	Różne	0,10	0,00	1,00	0,30	Różne
ModificationOfData	0,23	0,00	1,00	0,30	Różne	0,67	0,00	1,00	0,30	Różne
UnauthorizedAccess	0,63	0,00	1,00	0,30	Różne	0,48	0,00	1,00	0,30	Różne

Lemma B	UnauthorizedAccess				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
DenialOfServices	0,24	0,00	1,00	0,30	Różne
Destruction	0,22	0,00	1,00	0,30	Różne
Disclosure	0,13	0,00	1,00	0,30	Różne
ModificationOfData	0,30	0,00	1,00	0,30	Różne
UnauthorizedAccess	1,00	X	X	X	Tożsame

Lemma B	InformationLoss					LossOfInformationSystemAvailability				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
ThreatExploitsVulnerabilityOfAnAsset	0,26	0,00	0,50	0,15	Różne	0,50	0,00	0,50	0,15	Różne
UndesirableEvent	0,32	0,00	1,00	0,30	Różne	0,18	0,00	1,00	0,30	Różne
UnexpectedEvent	0,36	0,00	1,00	0,30	Różne	0,19	0,00	1,00	0,30	Różne

Lemma B	UnauthorizedDataModification					UnauthorizedDestructionOfInformation				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
ThreatExploitsVulnerabilityOfAnAsset	0,22	0,00	0,50	0,15	Różne	0,19	0,00	0,50	0,15	Różne
UndesirableEvent	0,26	0,00	1,00	0,30	Różne	0,24	0,00	1,00	0,30	Różne
UnexpectedEvent	0,29	0,00	1,00	0,30	Różne	0,27	0,00	1,00	0,30	Różne

Lemma B	UnauthorizedInformationDisclosure				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
ThreatExploitsVulnerabilityOfAnAsset	0,16	0,00	0,50	0,15	Różne
UndesirableEvent	0,24	0,00	1,00	0,30	Różne
UnexpectedEvent	0,27	0,00	1,00	0,30	Różne

Lemma B	ThreatExploitsVulnerabilityOfAnAsset					UndesirableEvent				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Impact	0,07	0,00	0,00	0,00	Różne	0,41	0,00	0,00	0,00	Różne

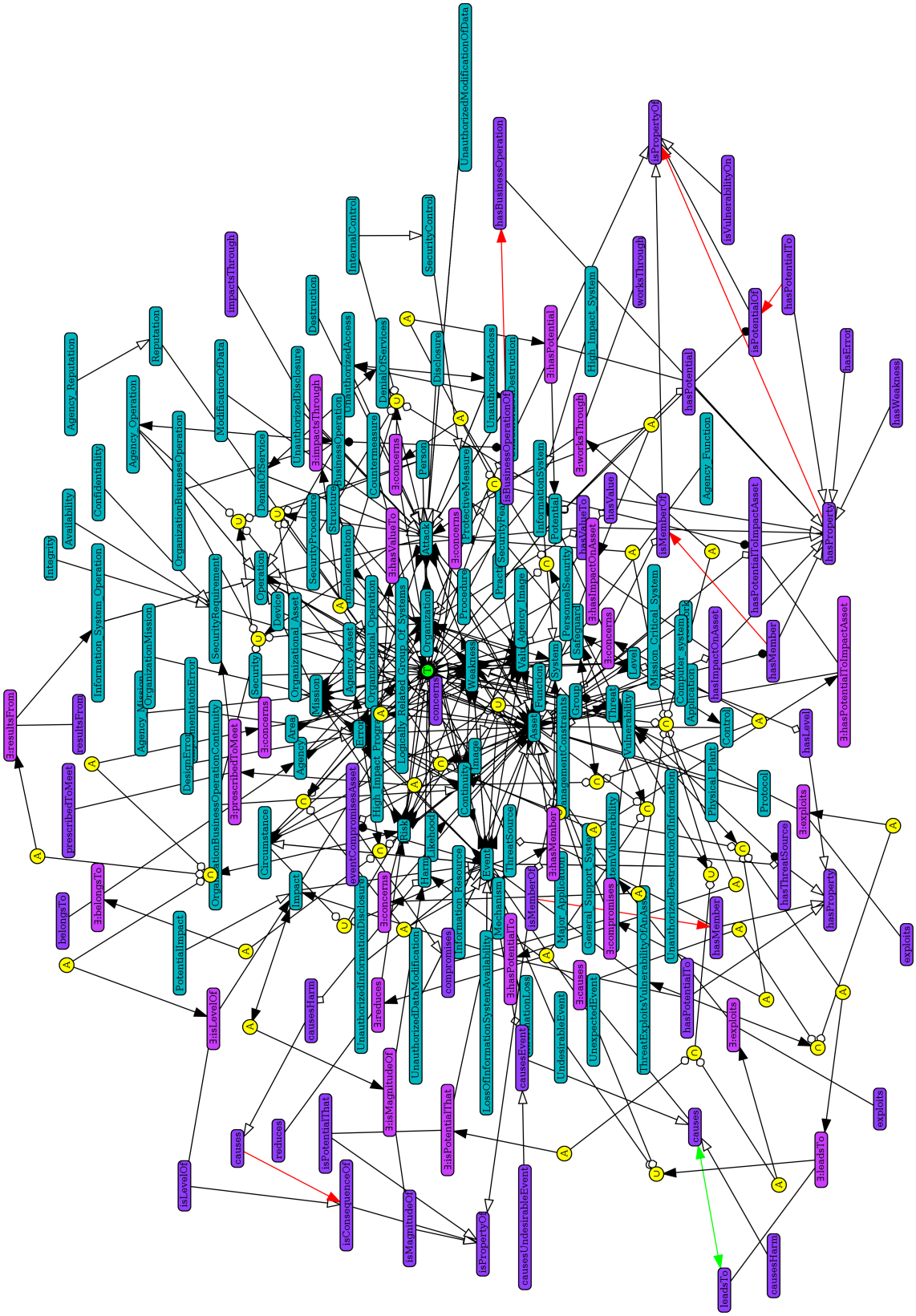
Lemma B	UnexpectedEvent				
Lemma A	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Impact	0,41	0,00	0,00	0,00	Różne

Tablica 4.5: Wyniki wzajemnych porównań pojęć dziedziczących po konceptach innych niż *owl:Thing*) obu integrowanych ontologii (c.d.)

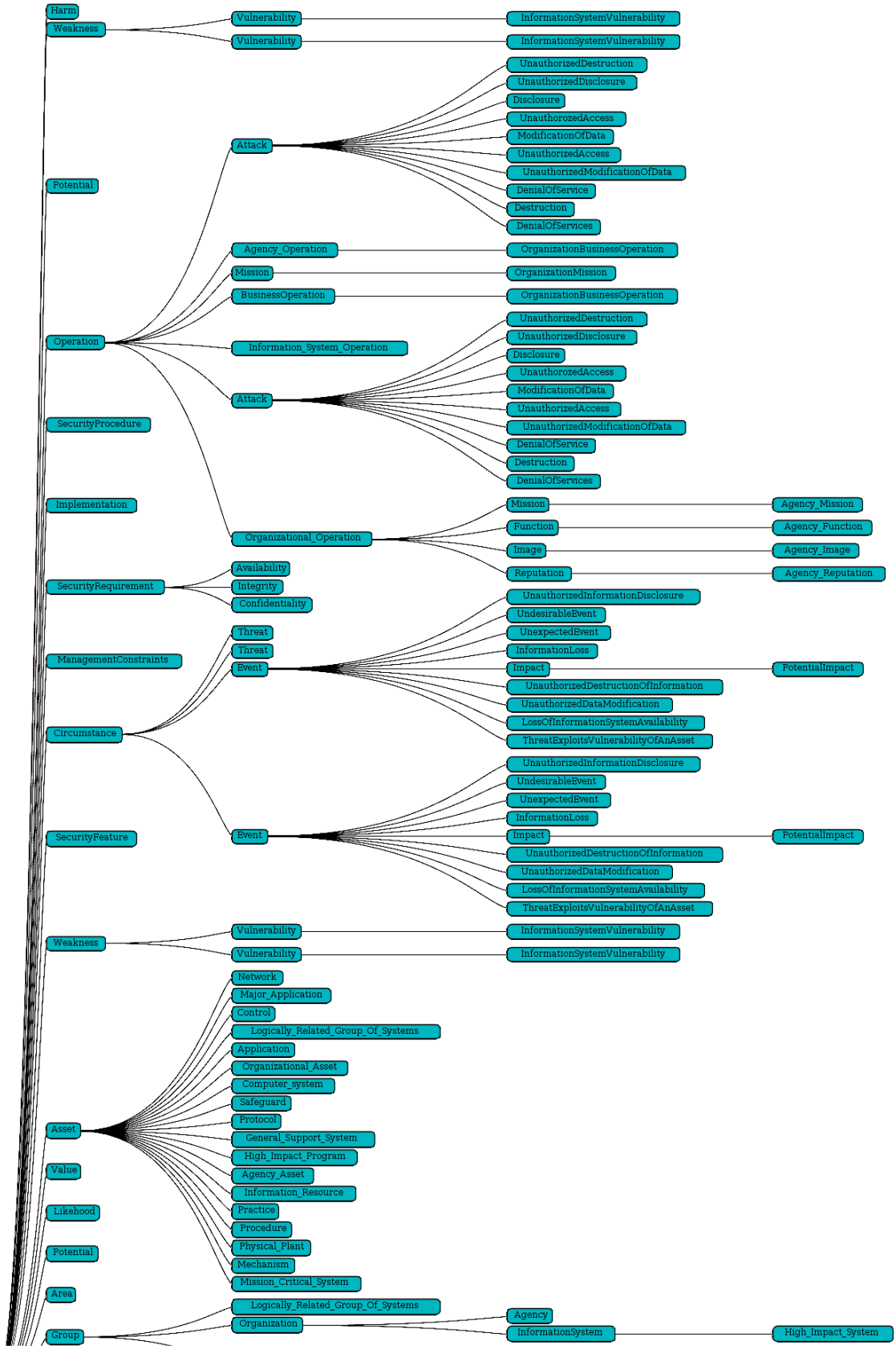
Lemma A	Lemma B	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
Attack	BusinessOperation	0,48	0,00	0,00	0,00	Różne
Organization	LogicallyRelatedGroupOfSystems	0,25	0,00	0,00	0,00	Różne
Vulnerability	Vulnerability	1,00	X	X	X	Tożsame

Algorytm analizuje integrowane ontologie warstwami od najbardziej ogólnych pojęć do najbardziej szczegółowych. Liczba porównań będzie więc w dużej mierze uzależniona zarówno od rozmiaru, jak i struktury łączonych ontologii. Im większe ontologie, tym więcej potencjalnych porównań należy wykonać pomiędzy ich elementami. Ponadto jeżeli ontologie posiadają płaską hierarchię pojęć, liczba porównań dodatkowo wzrośnie. W pesymistycznym przypadku, gdy każdy element ontologii A będzie podobny do każdego elementu ontologii B, liczba wykonanych porównań wynosić będzie $O(ab)$, gdzie a to liczba klas i bytów ontologii A, b to liczba klas i bytów ontologii B.

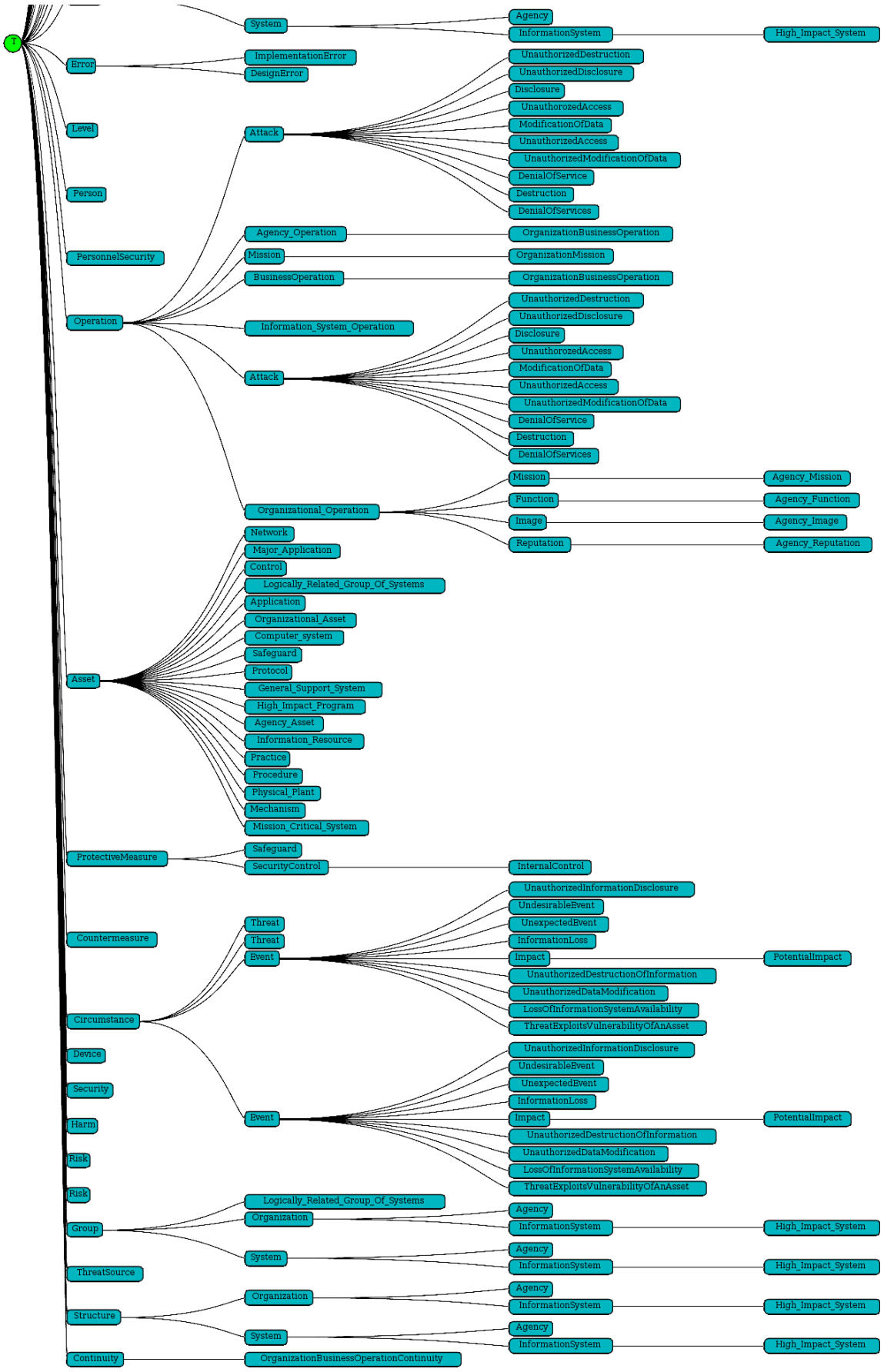
Należy zauważyć, że w celu zachowania oryginalnego nazewnictwa i struktury obu ontologii algorytm nie wykonuje fizycznego scalania konceptów. Łączenie elementów odbywa się za pomocą zdefiniowanego w standardzie OWL aksiomu *owl:equivalentClass*, mówiącego, że powiązane nim elementy ontologii są tożsame. Końcowy wynik działania algorytmu zaprezentowany jest na rys. 4.7, a wywnioskowana hierarchia nowo powstałej ontologii na rys. 4.8. Na pierwszym z obrazków, dla przejrzystości scalono tożsame koncepty o tych samych nazwach. Drugi z rysunków zawiera wszystkie klasy z obu ontologii. Jak można zauważyć na załączonych grafikach, wynikowa ontologia jest dużo większa niż ontologie wejściowe. Liczba konceptów wynosi tutaj sumę liczby pojęć ontologii wejściowych, a hierarchia wywnioskowana obrazuje ponadto liczne dodatkowe relacje wprowadzone przez algorytm. Takie działanie jest niezbędne w przypadku, gdy aplikacje rozróżniają koncepty nie tylko po nazwie, ale również na podstawie *URI*, będącym jednoznacznym identyfikatorem konceptu. Często jednak w operacjach na ontologiach istotne są tylko nazwy pojęć. W takich przypadkach w wynikowej ontologii można *URI* ujednolicić. Spowoduje to zmniejszenie liczby pojęć w przypadku, gdy w obu integrowanych ontologiach występują pojęcia o identycznych nazwach (rys. 4.9 oraz rys. 4.10).



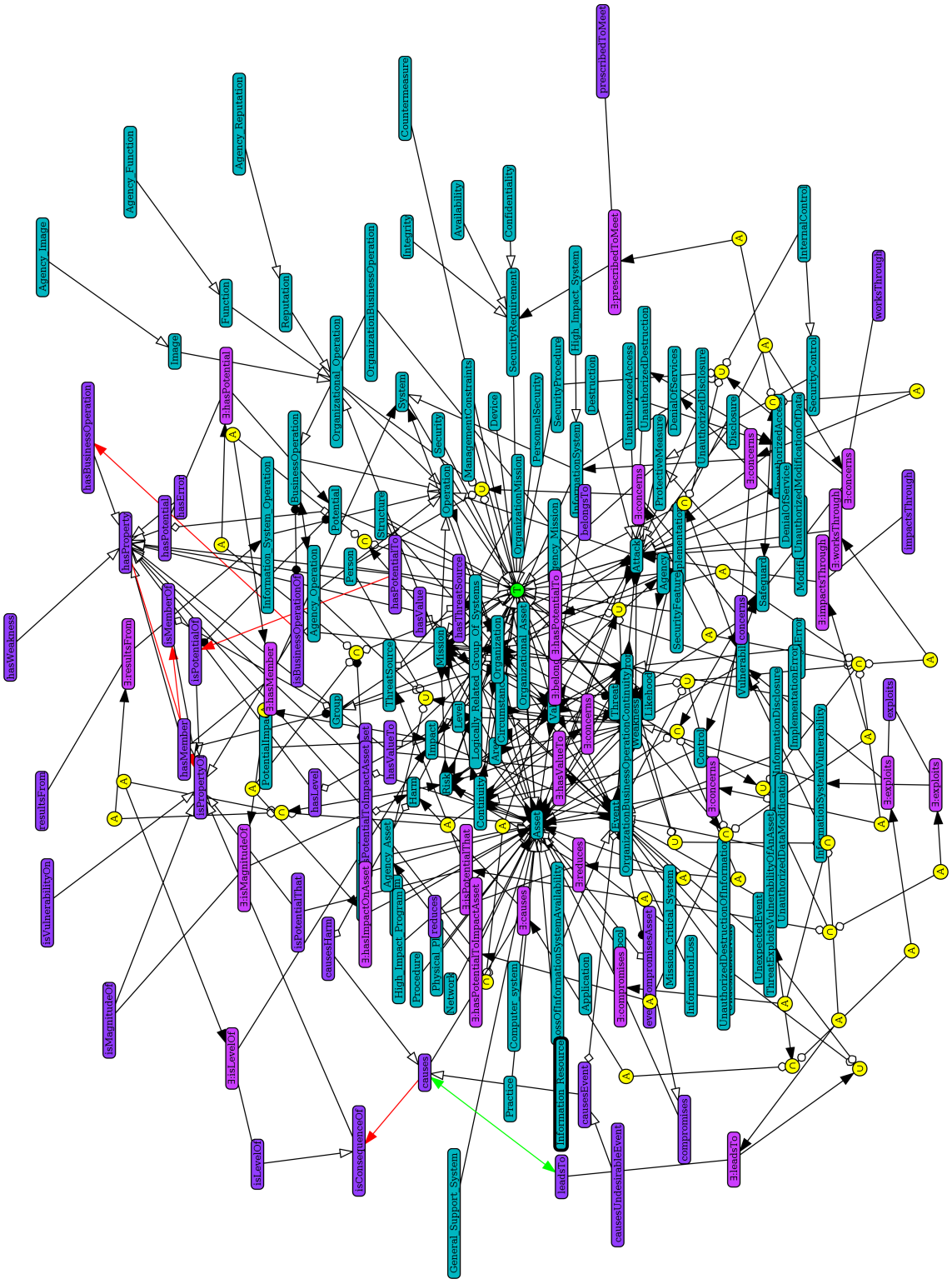
Rysunek 4.7: Konstrukcja wynikowej ontologii z zachowaniem URI – pełna ontologia



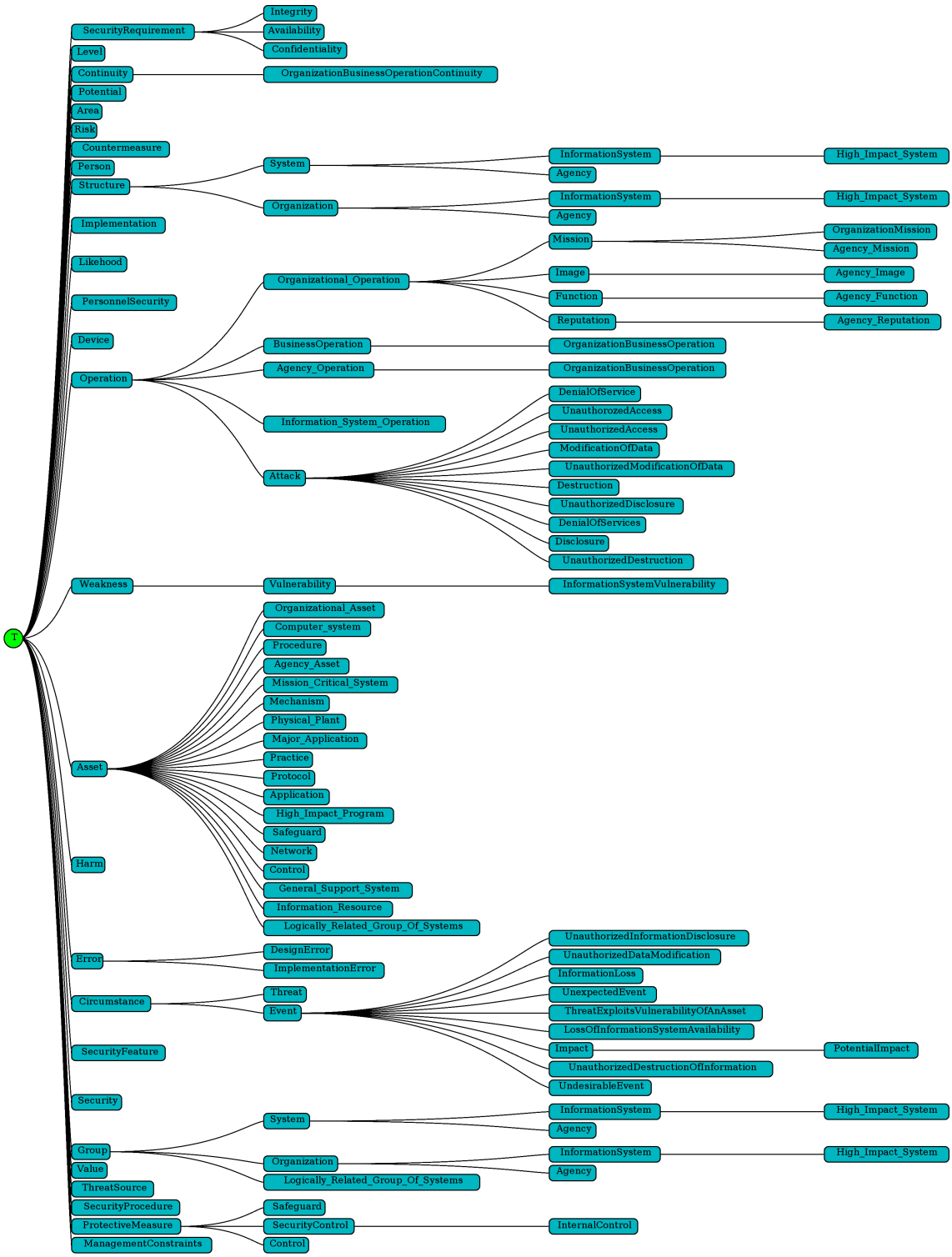
Rysunek 4.8: Konstrukcja wynikowej ontologii z zachowaniem URI – wywnioskowana hierarchia klas



Rysunek 4.8: (ciąg dalszy)



Rysunek 4.9: Konstrukcja wynikowej ontologii z nowym URI – pełna ontologia



Rysunek 4.10: Konstrukcja wynikowej ontologii z nowym URI– wywnioskowana hierarchia klas

Rozdział 5

Zastosowanie metod integracji ontologii w systemie OCS

5.1 Wprowadzenie

W tym rozdziale zaprezentowano autorski system OCS, służący do budowy, wersjonowania, składowania i integrowania ontologii. Zaprezentowano architekturę systemu i jego główne mechanizmy, takie jak metody konwersji reprezentacji ontologii, ich wersjonowanie, czy możliwość współpracy z edytorem Protégé.

W dalszej części rozdziału przedstawiono możliwości wizualizacji za pośrednictwem narzędzia SOVA oraz sposób działania modułu integracji ontologii, wykorzystującego zaproponowany algorytm.

System OCS stanowi podstawowe narzędzie wykorzystywane w trakcie analizy, opisanej w kolejnym rozdziale, jakości zaproponowanego rozwiązania.

5.2 OCS - Ontology Creation System

System OCS [18, 20, 21, 22], którego głównym konstruktorem jest autor niniejszej rozprawy, służy do konstruowania ontologii w środowisku rozproszonym umożliwiając uzgadnianie wspólnej warstwy konceptualnej. Opisywany system służy do zespołowego wytwarzania ontologii, a także ich składowania i udostępniania innym systemom.

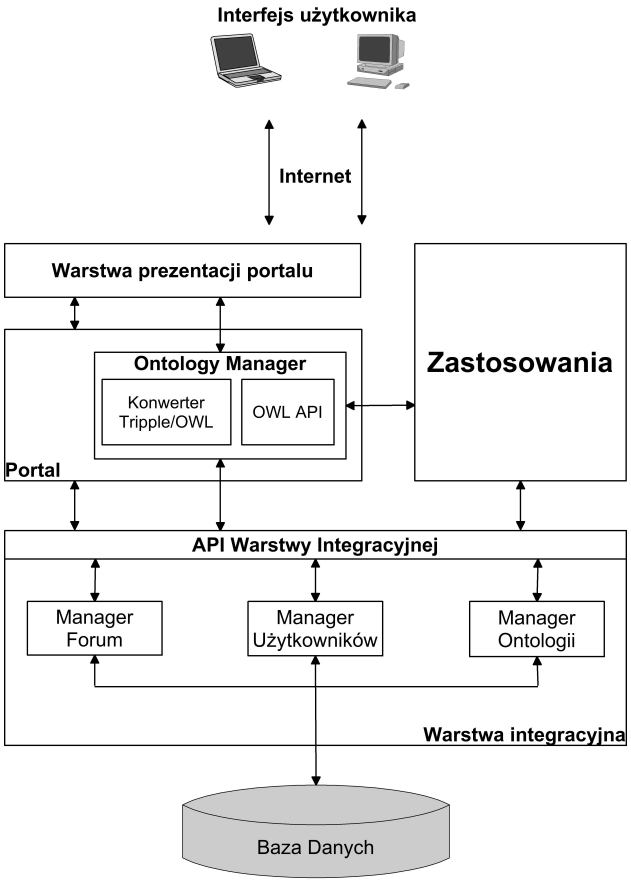
Podstawową funkcjonalnością systemu jest możliwość edycji ontologii, czyli tworzenia, modyfikacji oraz usuwania klas, atrybutów oraz relacji. Ponadto system umożliwia trwałe przechowywanie za jego pomocą tworzonych i modyfikowanych ontologii, a także podgląd i modyfikację na dowolnym etapie ich konstrukcji. Operacje te odbywają się dzięki zaimplementowanym mechanizmom wersjonowania podobnych do znanych z np. Subversion [4, 35], jednak operujących na poziomie semantycznym.

Ważnym aspektem systemu jest możliwość pracy zespołowej – narzędzie umożliwia pracę grupową nad ontologiami. Modyfikacje do ontologii wprowadzać może dowolny zarejestrowany użytkownik. Każda modyfikacja rejestrowana jest jako sugestie zmian, które z kolei muszą zostać zaakceptowane przez użytkowników odpowiedzialnych za stan danej ontologii. Dopiero po tym fakcie sugerowane zmiany stają się integralną częścią opracowywanego rozwiązania wchodząc w skład nowej wersji ontologii. Dzięki temu, pomimo pracy nad ontologią dowolnie dużej grupy ludzi, ostateczne rozwiązanie zachowuje spójność oraz minimalizowane jest ryzyko wprowadzenia fałszywych informacji. Proces korekty konfliktów pomiędzy propozycjami zmian wykonywany jest ręcznie przez autora ostatniej konfliktowej propozycji.

W przypadku pracy nad dużymi ontologiami przewidziano możliwość podziału prac między różnych ekspertów, zajmujących się jedynie fragmentami całego rozwiązania. Każdy z ekspertów rozpatruje jedynie swój fragment ontologii z właściwymi mu propozycjami zmian.

Drugim elementem wspomagającym pracę grupową nad ontologią jest forum dołączone do każdej z opracowywanych ontologii. Za jego pomocą użytkownicy systemu będą mogli prowadzić dyskusje nad kształtem poszczególnych elementów ontologii, czy kierunku, w jakim zmierzać powinno docelowe rozwiązanie.

5.3 Architektura systemu



Rysunek 5.1: Architektura systemu OCS

Ogólna architektura portalu zaprezentowana jest na rys. 5.1. System składa się z czterech głównych komponentów. Są to:

- Warstwa integracyjna wraz z bazą danych – odpowiedzialna za przechowywanie ontologii, oraz zarządzanie wersjami ontologii oraz użytkownikami systemu, steruje dostępem do danych przechowywanych w systemie. Ontologie przechowywane są w bazie danych w postaci zbioru trójek. Ten moduł dostarcza też całej logiki biznesowej związanej z zarządzaniem ontologiami oraz ich wersjami, dostępem do danych, zarządzaniem użytkownikami itp. Dostęp do przechowywanych w bazie danych odbywa się dwojako: bezpośrednio poprzez EJB lub WebServices oraz za pośrednictwem biblioteki OntologyManager. Zarejestrowany użytkownik ma możliwość pobrania dowolnej wersji każdej z ontologii i wykorzystania jej w dowolnym

zewnętrznym zastosowaniu. Warstwa integracyjna, poprzez mechanizmy tworzenia ontologii, może być wykorzystywana przez systemy automatycznego generowania ontologii na podstawie automatycznie przetwarzanych informacji o zewnętrznym świecie.

- OntologyManager – będący biblioteką przetwarzającą ontologię z postaci trójkowej (przechowywanej w bazie danych) do postaci obiektowej wykorzystywanej w samym edytorze, czyli obiektów OWL API [13, 73]. W dużej mierze odpowiada za komunikację z Warstwą integracji opakowując bezpośrednie wywołania poprzez EJB. Dzięki tej bibliotece możliwe jest korzystanie z zasobów repozytorium ontologii bez konieczności samodzielnego konwertowania reprezentacji ontologii z trójkowej na obiektową i odwrotnie. Umożliwia też bezpieczne zdalne połączenie z repozytorium dzięki zastosowaniu szyfrowania SSL do wszystkich przesyłanych komunikatów. Pozwala również na generowanie propozycji zmian, służących do zgłaszania poprawek do ontologii. Poprzez bibliotekę narzędziową dostępne jest całe API udostępniane przez warstwę integracyjną.

- Portal internetowy – strona WWW systemu oraz graficzny edytor ontologii uruchamiany jako aplikacja Java Web Start [116]. Umożliwia pracę offline jak i online – połączenie z siecią konieczne jest jedynie w momencie pobierania projektu z serwera oraz zgłaszania propozycji zmian do ontologii. Umożliwia wizualizację wytworzonej ontologii dzięki zastosowaniu biblioteki Prefuse [120]. Z edytorem zintegrowano również bibliotekę wnioskującą Pellet [127], co umożliwia graficzny podgląd pełnej, wywnioskowanej hierarchii rozszerzonej o elementy wyrażone nie wprost za pomocą właściwości obiektów i ich wzajemnych powiązań.

- Moduły zastosowań – będące elementami zewnętrznymi wykorzystującymi w praktycznych zastosowaniach ontologie wytworzone w systemie. Jako moduły zastosowań traktuje się wszystkie zewnętrzne systemy korzystające z repozytorium ontologii zarówno poprzez bezpośrednie wywołania EJB czy WebServices czy też za pośrednictwem Ontology Managera.

5.4 Konwersja reprezentacji ontologii

Edytor ontologii systemu OCS, podobnie jak pozostałe moduły, korzysta z biblioteki OWL API. Podczas pracy nad ontologią wszelkie operacje wykonywane są na jej obiektowej reprezentacji, natomiast przechowywanie ontologii możliwe jest dopiero po jej eksporcie. Jako że formaty XML, wspierane przez OWL API, słabo nadają się do przechowywania w bazach danych, podjęto decyzję o wykorzystaniu formatu trójkowego. W formacie tym ontologia jest listą trójek opisujących relację pomiędzy jej elementami. Podejście to pozwala na użycie relacyjnej bazy danych jako repozytorium ontologii – każda trójka może być pojedynczym wierszem w tabeli bazy danych. Rozwiązanie to pozwala też na łatwe generowanie i przechowywanie wspomnianych propozycji zmian, bez konieczności przechowywania całej zmienionej ontologii.

W związku z powyższym pojawiła się konieczność konwersji reprezentacji ontologii. Przy jej zapisie, przekształcana jest z postaci obiektowej, używanej w edytorze, do formatu trójkowego, używanego w bazie danych będącej repozytorium ontologii. Przy wczytywaniu ontologii z bazy danych do edytora konieczna jest konwersja odwrotna.

W trakcie odczytu ontologii z bazy pobierana jest lista trójek opisujących ontologię oraz dodatkowe elementy takie jak jej URI. Tworzona jest pusta ontologia w postaci obiektu obsługiwanego przez OWL API. Następnie trójki z listy są pojedynczo przekształcane na odpowiednie aksjomaty (ang. *axiom*), czyli relacje wiążące klasy, byty i właściwości wchodzące w skład ontologii, i dodawane do wcześniej utworzonego obiektu reprezentującego ontologię. Po przetworzeniu wszystkich trójek, obiekt ten zawiera pełną reprezentację ontologii i jest zwracany jako wynik parsowania.

Konwersja odwrotna przebiega analogicznie. Z obiektowej reprezentacji ontologii kolejno odczytywane są aksjomaty, a następnie generowana jest ich postać trójkowa. W wyniku tej operacji ontologia zwracana jest w postaci listy trójek odpowiadającym wszystkim aksjomatom, a następnie zapisywana w bazie danych.

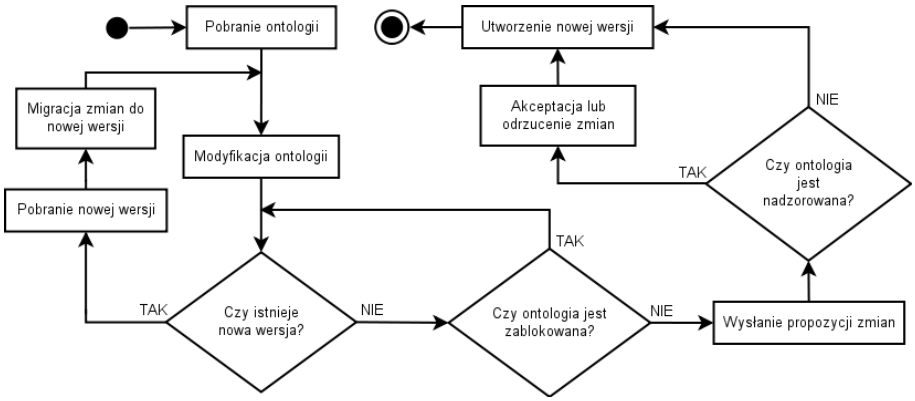
Dzięki trójkowemu formatowi reprezentacji ontologii możliwa jest efektywna implementacja tworzenia, przechowywania, pobierania i wprowadzania zmian ontologii. Tworzenie propozycji odbywa się poprzez porównanie trójkowej reprezentacji ontologii przed i po wprowadzeniu zmian. Sprowadza się ono do porównania ich w celu ustalenia, które trójki zostały dodane, a które usunięte. W wyniku tej operacji powstaje lista trójek wzbogacona o informację o konieczności usunięcia bądź dodania danej trójki z reprezentacji ontologii w celu wprowadzenia zmian. Propozycje reprezentowane w tym formacie są łatwe do przechowania w repozytorium, przygotowanym do obsługi trójek.

Poprzez zapis trójkowy jest również możliwe pobieranie zgłoszonych propozycji zmian oraz prezentowanie ich na obecnie wczytanej ontologii. Trójki te dzielone są na grupę trójek, które należy dodać do ontologii, oraz takich, które należy usunąć z ontologii. Na ich podstawie generowane są odpowiadające im aksjomaty w postaci obiektów OWL API. W kolejnym kroku są one umieszczane na listach AddAxiom i DeleteAxiom. Na tej podstawie, korzystając z metod OWL API, możliwe jest zmodyfikowanie bieżącej ontologii tak, by odzwierciedlała wybrane, bądź wszystkie, zmiany zasugerowane przez użytkowników systemu.

5.5 Praca grupowa nad ontologią

Aby możliwe było sprawne przetwarzanie wiedzy przez wszystkich potencjalnych odbiorców, konieczne jest, by reprezentowała ona wspólne spojrzenie na dane zagadnienie. Człowiek charakteryzuje się jednak mocno zindywidualizowanym poglądem na otaczającą go rzeczywistość. Stąd konieczne jest, by praca nad ontologią przebiegała w jak najszerszym gronie, pozwalając tym samym na ujednolicenie wizji opisu wybranego fragmentu świata.

System OCS opiera się na założeniu, że w sytuacji, gdy ontologia ma być stosowana przez szerokie gremium odbiorców, proces jej powstawania również powinien obejmować szeroką grupę pomysłodawców [22]. Zakładając system głosowania przyjęty w Collaborative Protégé [131] autor ontologii może łatwo stracić kontrolę nad kierunkiem jej tworzenia a ostateczny efekt może całkowicie odbiegać od przyjętych założeń. Stąd w proponowanym rozwiązaniu uprzywilejowany użytkownik, będący twórcą ontologii, lub ich grupa nominowana przez twórcę ontologii ma prawo akceptować zmiany w ontologii. Każdy inny użytkownik ma prawo zgłaszać propozycje zmian do dowolnej z ontologii, jednak nie zostaną one uwzględnione aż do etapu akceptacji zmian.



Rysunek 5.2: Proces zgłaszania propozycji zmian oraz tworzenia nowych wersji ontologii w systemie OCS

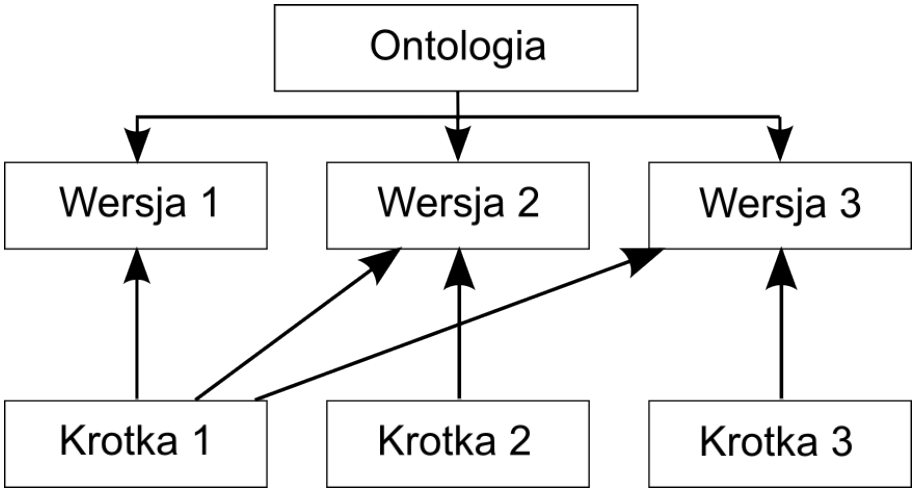
W przypadku ontologii publicznych, których rozwój nie jest nadzorowany, proces zgłaszania propozycji może być również połączony z tworzeniem nowej wersji, realizując algorytm znany z repozytoriów kodu, np. Subversion [10] ale w ujęciu semantycznym. W tym przypadku każdy użytkownik ma prawo tworzenia nowej wersji, a zgłoszone przez niego zmiany są automatycznie zatwierdzane.

Oba powyższe procesy zostały przedstawione na rys. 5.2. Należy pamiętać, że podobnie jak to ma miejsce w przypadku repozytoriów kodu, to użytkownik tworzący nową wersję odpowiedzialny jest za zachowanie spójności ontologii.

5.6 Wersjonowanie ontologii

Kolejnym istotnym aspektem pracy grupowej nad ontologią jest możliwość śledzenia zachodzących w niej zmian oraz dostęp do dowolnej wcześniejszej wersji ontologii. Obie te operacje są możliwe dzięki zastosowaniu języka OWL, a tym samym oparciu konstrukcji ontologii na trójkach RDF [14, 29].

W systemie ontologie składowane są jako trójki pogrupowane w poszczególne wersje (rys. 5.3). W tym modelu raz zapisana krotka pozostaje w systemie aż do usunięcia ontologii.



Rysunek 5.3: Sposób składowania ontologii w systemie OCS

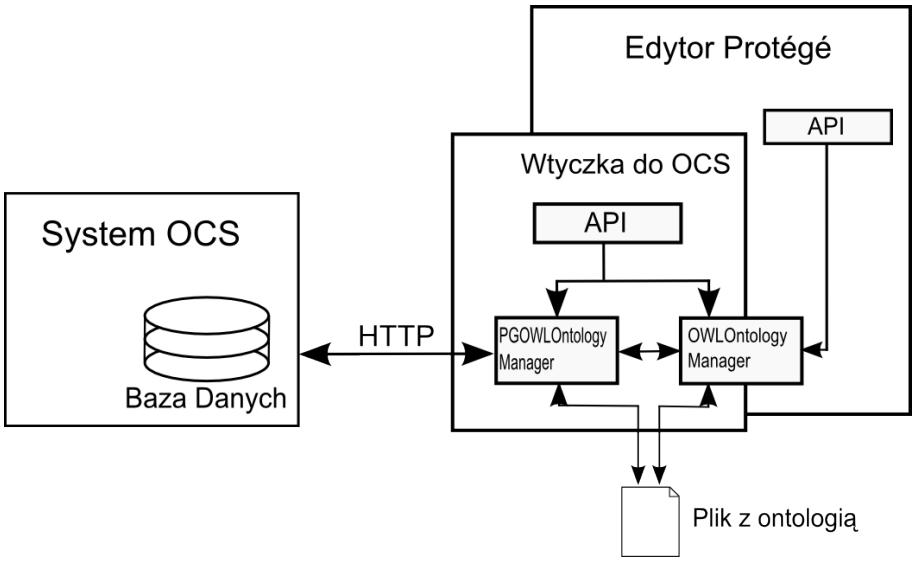
Umożliwia to bezpośrednie uzyskanie dostępu do dowolnej z wersji ontologii bez konieczności przetwarzania danych historycznych. Również porównanie dwu wersji ontologii na poziomie semantycznym wymaga jedynie porównania dwóch zbiorów trójek i określeniu różnic pomiędzy nimi. Wynik takiego porównania może zostać następnie przekształcony w aksjomaty języka OWL.

5.7 Zgodność z edytorem Protégé

Autor, chcąc skorzystać z bogatej funkcjonalności oraz popularności edytora Protégé, rozszerzył jego funkcjonalność o możliwość pracy grupowej opartej na przepływie zadań opracowanym dla systemu OCS. Rozszerzenie to obejmuje funkcjonalność pobierania dowolnej wersji ontologii z serwera i wysyłania propozycji zmian. Są to czynności niezbędne dla zwykłego użytkownika modyfikującego ontologię. Poprzez edytor Protégé właściciel ontologii ma możliwość dodawania nowych ontologii i tworzenia ich kolejnych wersji.

Programiści Protégé umożliwili dodawanie nowych funkcji do edytora poprzez moduł obsługi „wtyczek" (ang. *plugin*) [90]. Ponadto, zarówno system OCS jak i edytor Protégé, do operowania na ontologiach wykorzystują bibliotekę OWL API. Biblioteka ta dostarcza manager ontologii, pozwalający m.in. na wczytywanie i zapis plików ontologii. Pozwala również na edycję składowych wczytanych ontologii. Klasa managera w systemie OCS (PGOWLOntologyManager) jest rozszerzeniem klasy z biblioteki OWL (OWLOntologyManger) o funkcje pozwalające na komunikację z serwerem składującym i wersjonującym ontologie. Zezwala również na zarządzanie i uwierzytelnianie użytkowników, pobieranie ontologii o zadanym URI, porównywanie ontologii, czy też

wysyłanie propozycji zmian. Przestawioną architekturę wtyczki zaprezentowano na rys. 5.4.



Rysunek 5.4: Architektura wtyczki integrującej edytor Protégé z systemem OCS

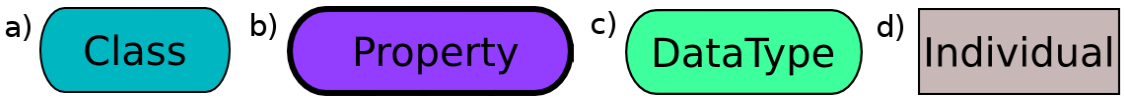
W celu zachowania kompatybilności wtyczki z systemem OCS interfejs użytkownika został przeniesiony z systemu OCS. Zachowano również zgodność plików projektu, dzięki temu możliwe jest edytowanie tego samego projektu zarówno w Protégé, jak i w edytorze OCS.

5.8 Wizualizacja ontologii

Proces wytwarzania ontologii jest procesem złożonym. Ilość elementów wchodzących w skład nawet niewielkiej ontologii może być bardzo duża. Aby wspomóc ten proces, na potrzeby systemu OCS utworzono bibliotekę wizualizacji ontologii o nazwie SOVA (Simple Ontology Visualization API) [21]. Biblioteka ta umożliwia pełną wizualizację złożonych ontologii w postaci grafu, którego wierzchołkami są klasy, byty oraz właściwości, a krawędziami relacje zachodzące pomiędzy tymi składowymi.

Biblioteka SOVA miała na celu umożliwić wizualizację wszystkich elementów wchodzących w skład ontologii, co umożliwia szybsze wychwycenie błędów i niedoróbek wynikających z użycia niepoprawnych konstrukcji języka OWL. Przygotowano więc zestaw symboli graficznych odpowiadających każdemu możliwemu elementowi w dilekcie DL języka OWL.

Symbole reprezentujące klasy (ang. *Class*), właściwości (ang. *Property*) oraz typy danych (ang. *DataType*) mają kształt prostokąta z zaokrąglonymi krawędziami, gdyż wszystkie one współdzielą podobne rodzaje relacji. W celu dalszego uproszczenia identyfikacji typu elementu różnią się one kolorem oraz promieniem zaokrąglenia wierzchołka. Byty (ang. *Individual*), jako elementy wchodzące w odmienne relacje, posiadają ostre wierzchołki. Symbole te prezentuje rys. 5.5.

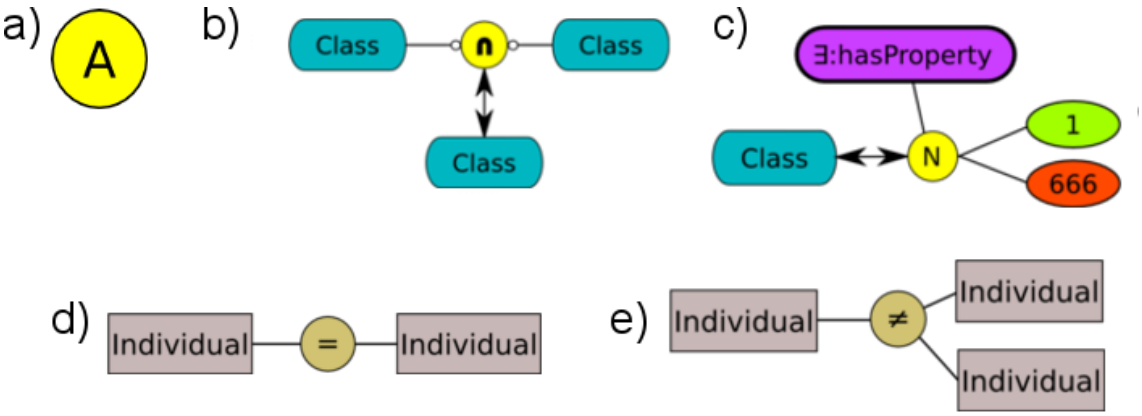


Rysunek 5.5: Symbole reprezentujące klasy (a), właściwości (b), typy danych (c) oraz byty (d)

Największym wyzwaniem przy wizualizacji ontologii jest prezentacja klas anonimowych. W przypadku biblioteki SOVA, wszelkie relacje wprowadzające klasy anonimowe przedstawiane są jako

dodatkowy wierzchołek w postaci żółtego koła z wpisaną literą „A", gdy jest to niezależna klasa anonimowa, lub graficznym symbolem precyzującym typ relacji tworzącej klasę anonimową (rys. 5.6). W przypadku przecięcia (ang. *intersection*), komplementarności (ang. *complementarity*), unii (ang. *union*) oraz liczności (ang. *cardinality*) zastosowano ogólnie znane symbole matematyczne, czyli odpowiednio: $\bigcap, \neg, \bigcup, N$. W przypadku relacji liczności dodatkowe wierzchołki reprezentują wartość minimalnej, maksymalnej lub dokładnej liczności (ang. *min*, *max* oraz *equal cardinality*).

Analogicznie postąpiono w przypadku relacji zachodzących pomiędzy bytami, z tym że koła mają odmienny kolor. Symbol z wpisanym znakiem = reprezentuje relację tożsamości (ang. *sameAs*), a $\neq$ relację rozłączności, zarówno w wersji jeden do jeden (ang. *differentFrom*) jak i wiele do wiele (ang. *allDifferent*).



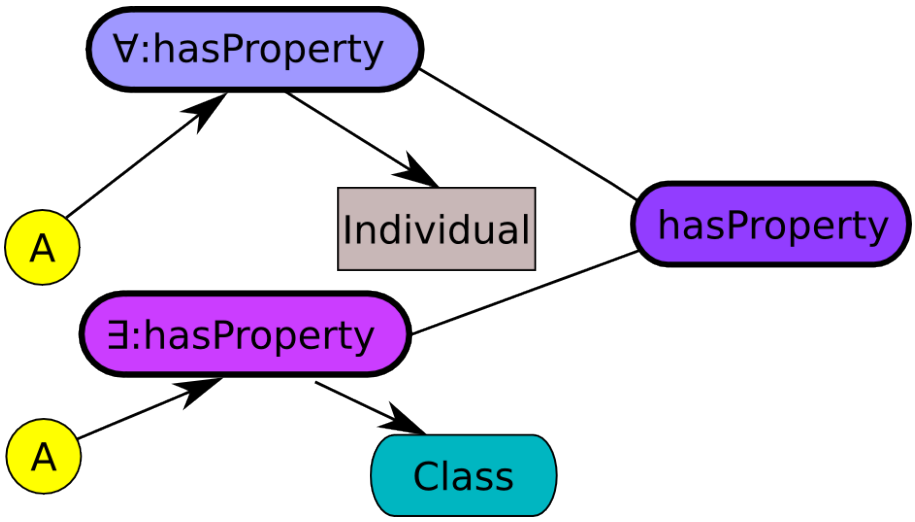
Rysunek 5.6: Przykładowe symbole prezentujące klasy anonimowe (a), przecięcie (b), minimalną i maksymalną liczność (ang. *cardinality*) (c), relację tożsamości (d) oraz relację rozłączności (e)

Klasy anonimowe są również wykorzystywane do zobrazowania relacji „owl:-allValuesFrom" oraz „owl:someValuesFrom" (rys. 5.7). Relacja „owl:allValues-From" zdefiniowana jest następująco: jeżeli dana jest klasa bytów x, dla których para (x, y) jest instancją właściwości P opisanej tą relacją, to y powinno być wystąpieniem klasy będącej przeciwdziedziną właściwości lub typem danych. Z kolei relacja „owl:someValuesFrom" definiuje klasę bytów x, dla których istnieje przynajmniej jeden y, będący instancją klasy lub wartością, dla którego para (x, y) jest instancją właściwości P opisanej tą relacją. Klasa anonimowa łączona jest krawędzią z wykorzystywaną właściwością.

Klasa będąca dziedziną właściwości połączona jest z nią strzałką wychodzącą od klasy i wskazującą na właściwość. W zależności od relacji nazwa właściwości poprzedzana jest kwantyfikatorem ogólnym (dla „owl:allValuesFrom") lub szczegółowym (dla „owl:someValuesFrom"). Strzałka prowadząca od właściwości wskazuje jej przeciwdziedzinę, która może być klasą, zarówno prostą jak i dowolnie złożoną reprezentowaną przez klasę anonimową, lub bytem. Dodatkowo, w celu zachowania spójności grafu, właściwości połączone są z ich definicjami linią ciągłą.

Relacje proste są w SOVA reprezentowane za pomocą linii z różnymi grotami. W ogólności kierunek strzałek wskazuje kierunek czytania danego aksjomatu. Tam, gdzie kierunek nie jest istotny, groty zostały pominięte, by zwiększyć czytelność wizualizacji. Niektóre relacje zaprezentowano na rys. 5.8.

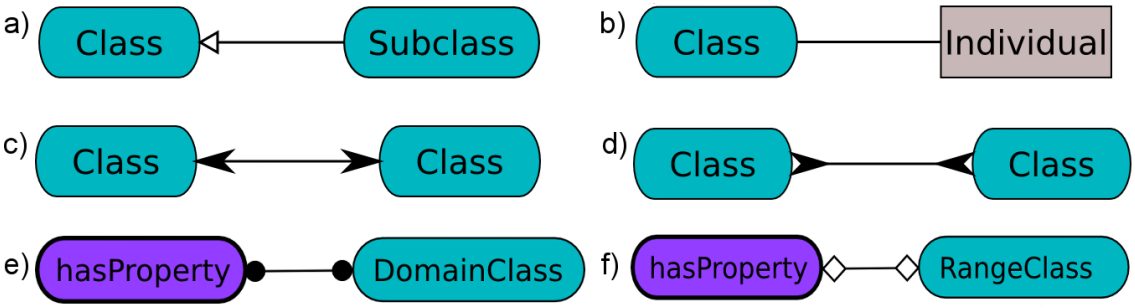
Tam, gdzie to było możliwe, wykorzystano istniejącą notację języka UML. Relacje dziedziczenia klas („subClass") oraz właściwości („subProperty") reprezentowane są poprzez strzałki z pustymi grotami. Tożsamość (ang. *equivalent*) oraz rozłączność (ang. *disjoint*) reprezentowane są poprzez linie ze strzałkami zwróconymi w przeciwne strony w celu podkreślenia ich przeciwstawności. W przypadku definicji właściwości, odwrotność oznaczona jest poprzez czerwony kolor strzałki. Rodzaj strzałki uzależniony jest od symetryczności danej relacji. Tożsamość właściwości



Rysunek 5.7: Przykładowa reprezentacja relacji „someValuesFrom” oraz „allValuesFrom”

odróżniona jest od tożsamości klas innym kolorem strzałki.

Dziedzinę (ang. *domain*) oraz przeciwdziedzinę (ang. *range*) właściwości dodatkowo zaprezentowano jako osobne groty (rys. 5.8 e oraz f).



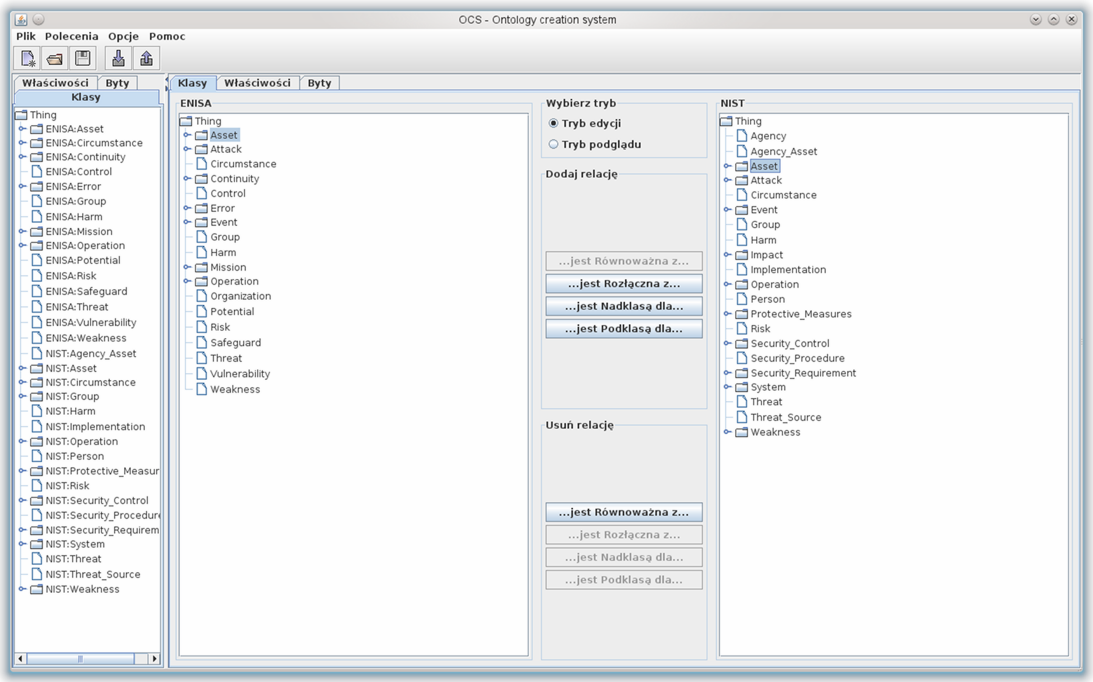
Rysunek 5.8: Przykładowa reprezentacja relacji: *rdfs:subclassOf* (a), *instanceOf* (b), *owl:equivalentClass* (c), *owl:disjointWith* (d), *rdfs:domain* (e) oraz *rdfs:range* (f)

5.9 Łączenie ontologii

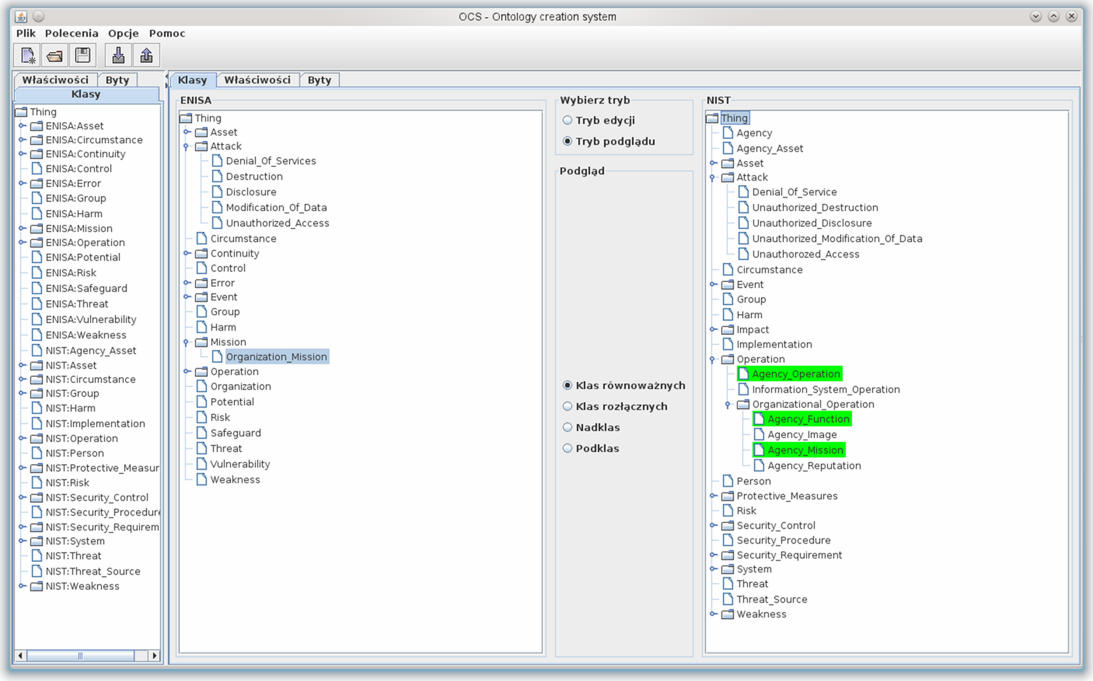
System OCS, w jednej z ostatnich faz rozwoju, rozbudowany został o możliwość integracji ontologii. Interfejs użytkownika został rozszerzony o możliwość prezentacji trzech ontologii: dwóch integrowanych (centralna i prawa część interfejsu) oraz wynikową (lewa część interfejsu). Przykładowy zrzut ekranu interfejsu modułu integracji zaprezentowano na rysunku 5.9. Każda z ontologii prezentowana jest w postaci drzewa znanego np. z edytora Protégé.

Pomiędzy dwoma integrowanymi ontologiami znajdują się przyciski, umożliwiające szybkie dodanie lub usunięcie relacji wiążących elementy obu ontologii. Zestaw przycisków zmienia się w zależności od typu elementów (klas, bytów i właściwości). Odpowiednie przyciski (dodawania lub usuwania relacji) są włączane i wyłączane w zależności od tego, czy dana relacja została już dodana czy nie. Analogicznie włączane i wyłączane są przyciski umożliwiające usunięcie wskazanej relacji.

Przełączenie w tryb podglądu umożliwia z kolei pokazanie wszystkich relacji, w jakich wybrany element jednej z ontologii znajduje się względem elementów drugiej ontologii (rys. 5.10).



Rysunek 5.9: Interfejs modułu integracji ontologii systemu OCS – tryb edycji relacji

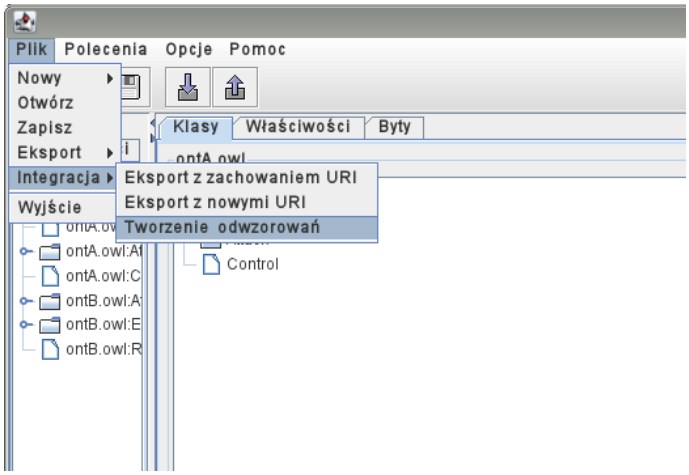


Rysunek 5.10: Interfejs modułu integracji ontologii systemu OCS – tryb podglądu relacji

W przypadku, gdy zachodzi konieczność dodania lub zmodyfikowania któregoś z elementów wynikowej ontologii, lub wprowadzenia innej, bardziej złożonej relacji, użytkownik może to zrobić bezpośrednio korzystając z menu kontekstowego drzewa zawierającego wynikową ontologię.

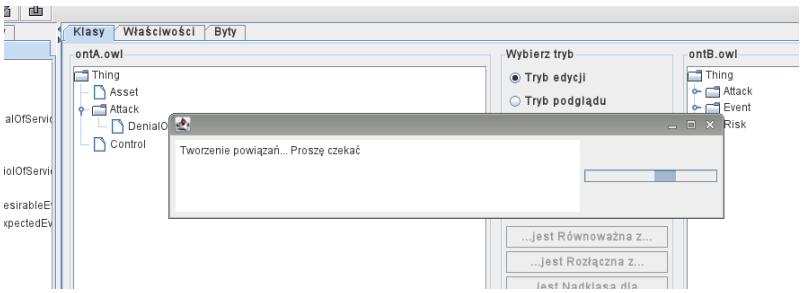
Finalną ontologię można zapisać w osobnym pliku na dwa sposoby. Jeden z nich to zachowanie oryginalnych URI obu ontologii i korzystanie z mechanizmu importowania dostępnego w języku OWL, tworząc tym samym odwzorowanie elementów jednej ontologii w drugą. Drugi sposób polega na eksporcie połączonej ontologii oraz zmianie URI ontologii źródłowych na URI ontologii wynikowej.

System OCS został również rozbudowany o implementację proponowanego w niniejszej rozprawie algorytmu. Integrując dwie ontologie użytkownik może wygenerować powiązania pomiędzy ich elementami wywołując algorytm poprzez menu „Plik” –> „Integracja” –> „Tworzenie Odwzorowań” (rys. 5.11).



Rysunek 5.11: Interfejs modułu integracji ontologii systemu OCS – aktywacja algorytmu

Po potwierdzeniu algorytm rozpoczyna pracę. W jej wyniku zastąpione zostaną wszystkie dotychczasowe powiązania nowymi wygenerowanymi przez algorytm. System OCS na bieżąco informuje o wykonywanych czynnościach, gdyż w przypadku dużych ontologii wywołanie algorytmu może zająć dużo czasu (rys. 5.12).



Rysunek 5.12: Interfejs modułu integracji ontologii systemu OCS – informacja o działaniu algorytmu

Rozdział 6

Analiza jakości algorytmu

6.1 Wprowadzenie

Analizę jakości proponowanego algorytmu oparto na wybranych ontologiach opracowanych w ramach EON Ontology Alignment Contest [51] oraz konstrukcji ontologii bezpieczeństwa. W pierwszym przypadku zachowanie proponowanego algorytmu przetestowano na rzeczywistych ontologiach stosowanych przez organizację Ontology Alignment Evaluation Initiative i porównano z referencyjnymi ontologiami zaprezentowanymi przez autorów konkursu. W drugim przypadku opracowano ontologię bezpieczeństwa poprzez konstrukcję, a następnie integracje mniejszych ontologii. Proces ten został przeprowadzony zarówno ręcznie przez człowieka, jak i półautomatycznie za pomocą proponowanego algorytmu.

W pierwszej części rozdziału zachowanie proponowanego algorytmu przetestowano na rzeczywistych ontologiach stosowanych przez organizację Ontology Alignment Evaluation Initiative. Przeprowadzono analizę zachowania algorytmu w przypadku łączenia wzorcowej ontologii z:

- identyczną ontologią,
- całkowicie odmienną ontologią,
- ontologiami podobnymi zapisanymi w bardziej ogólnych dialektach języka OWL,
- identyczną ontologią pozbawioną znaczących etykiet pojęć.

Efekty działania algorytmu porównano z referencyjnymi ontologiami zaprezentowanymi przez autorów konkursu.

W drugiej części rozdziału zaprezentowano opracowaną ontologię bezpieczeństwa oraz proces jej konstrukcji. Porównano efekty działania algorytmu z wynikami ręcznej integracji modułów wchodzących w skład proponowanej ontologii.

W ostatniej części rozdziału przedstawiono ocenę działania algorytmu oraz możliwości jego zastosowania w systemach informatycznych.

6.2 Ontologie opracowane w ramach EON Ontology Alignment Contest

W związku z rosnącą liczbą badań nad łączeniem i odwzorowywaniem ontologii, organizacja Ontology Alignment Evaluation Initiative (OEIA) od 2004 roku organizuje warsztaty badające jakość pozyskanych rozwiązań [51]. W trakcie warsztatów opracowano szereg ontologii i przypadków testowych, z których część została zaadaptowana na potrzeby weryfikacji poprawności proponowanego

algorytmu. Wszystkie wykorzystane w niniejszym rozdziale ontologie są publicznie dostępne pod adresem: `http://oaei.ontologymatching.org/2006/benchmarks/`. Niektóre ontologie musiały zostać nieznacznie zmodyfikowane w celu usunięcia niezgodności z nowszymi wersjami biblioteki OWL API oraz wnioskera HermiT. Wszystkie ewentualne zmiany zostały podane w podrozdziałach opisujących dotyczące je testy.

6.2.1 Wyjściowa ontologia

Jako podstawowa ontologia wykorzystana została ontologia opisująca strukturę znaczników spisów bibliograficznych w zgodnych z semantyką BibTeX. Jest to niewielka ontologia składająca się z 37 klas i 57 instancji, napisana w dialekcie OWL-DL. Jako reprezentację plikową zastosowano format RDF/XML. Zaprezentowana została na rys. 6.1 oraz rys. 6.2.

Ontologię tę można pobrać spod adresu: `http://oaei.ontologymatching.org/2006/benchmarks/101/onto.rdf`. Wszelkie kolejne testy polegać będą na próbie integracji danej ontologii z zaprezentowaną w tym rozdziale.

6.2.2 Łączenie identycznych ontologii

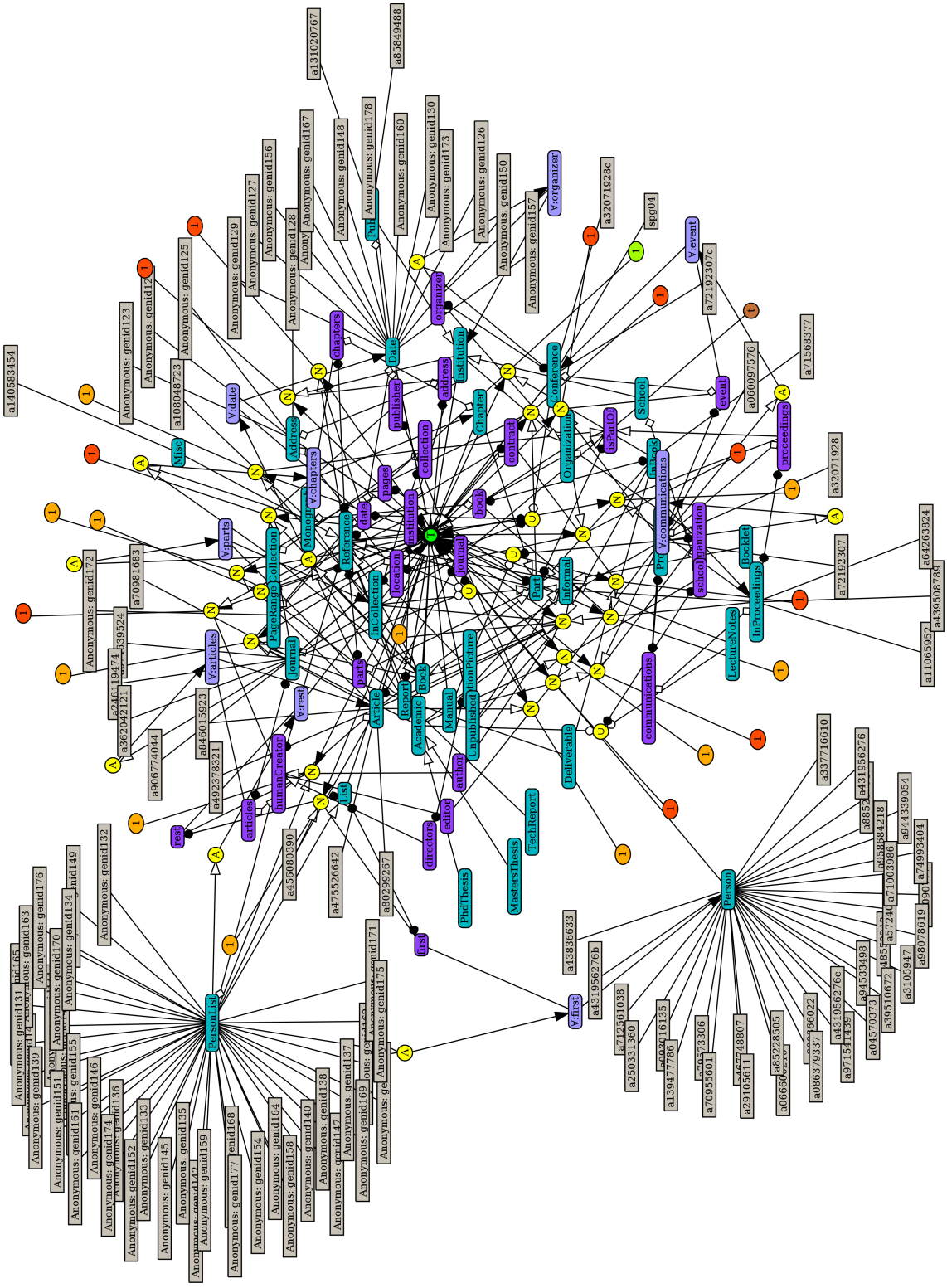
Ontologię zaprezentowaną w sekcji 6.2.1 połączono z nią samą. Żadna z ontologii nie była modyfikowana, zmianie uległ jedynie identyfikator URI, co umożliwiło jednoczesne wczytanie obu plików za pośrednictwem biblioteki OWL API. W wyniku działania algorytmu uzyskano ontologie identyczną z wejściową z dodatkowym powiązaniem pomiędzy klasami „Adres" (ang. *Address*) oraz Referencja (ang. *Reference*). Wszystkie powiązania utworzone zostały na podstawie podobieństwa wynoszącego 1.0.

Dodatkowe powiązanie wynika z faktu, że w dziedzinie informatyki wyrazy te są synonimami. Testowa ontologia nie dostarcza niestety żadnych dodatkowych wskazówek pozwalających na zawężenie znaczenia słów poprzez np. definicję rozłączności klas.

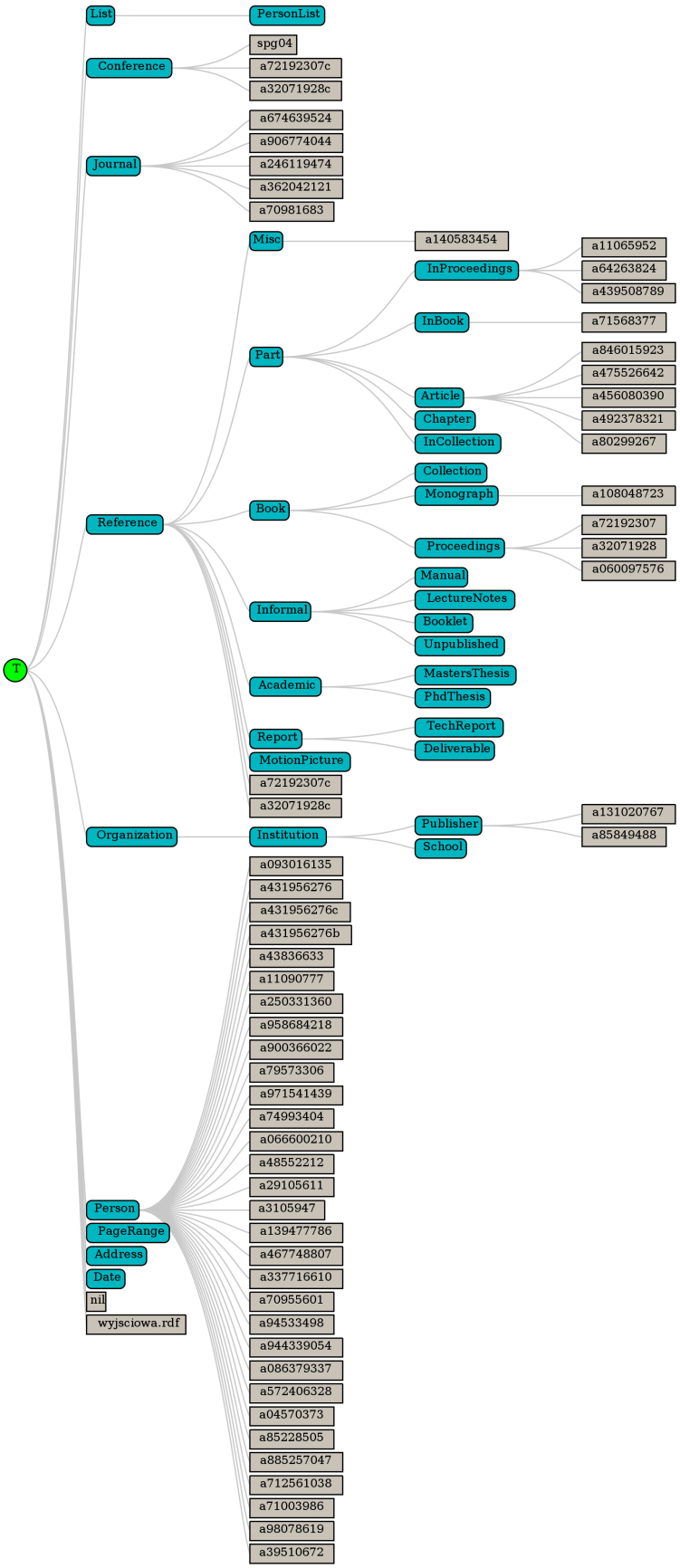
6.2.3 Łączenie niepowiązanych ze sobą ontologii

W drugim kroku ontologię zaprezentowaną w sekcji 6.2.1 połączono z ontologią win i posiłków dostępna pod adresem `http://oaei.ontologymatching.org/2006/benchmarks/102/onto.rdf`. Jest to mała ontologia, składającą się z 77 klas i 161 bytów. Z ontologii tej należało usunąć jeden ze znaczników importu ontologii, gdyż jej konstrukcja powodowała występowanie zależności cyklicznych poprzez ładowanie ontologii zależnej on niej samej. Usunięcie deklaracji importu nie zmienia konstrukcji ontologii, gdyż załączana za jej pośrednictwem ontologia dołączona zostanie pośrednio przez inną importowaną ontologię.

W wyniku pracy algorytmu powstała trzecia ontologia będąca sumą obu wejściowych ontologii. Algorytm wykazał jedynie podobieństwo semantyczne pomiędzy pojęciami „Date" („Data") oraz „Grape" („Winogrono"). Zastosowana tu miara Levenshteina wskazała $P_{sem} = 0,77$. Analiza struktury słownika WordNet wykazała jednak niezależność tych pojęć. Algorytm połączył więc obie ontologie poprawnie.



Rysunek 6.1: Ontologia reprezentująca atrybuty bibliografii opartej na BibTeX – pełna ontologia



Rysunek 6.2: Ontologia reprezentująca atrybuty bibliografii opartej na BibTeX – hierarchia wywnioskowana

6.2.4 Uogólnienie języka wyrazu

Kolejne dwa testy integrują bazową ontologię (sekcja 6.2.1) z analogicznymi zapisanymi w dialekcie OWL Lite języka OWL. W pierwszym przypadku (rys. 6.3) relacje niedostępne w dialekcie OWL Lite zostały zastąpione zbliżonymi, bardziej ogólnymi. W ogólności usunięte zostały wszystkie relacje typu *owl:unionOf* oraz *owl:oneOf* oraz typy właściwości *owl:TransitiveProperty*. W drugim przypadku (rys. 6.4) brakujące relacje zostały usunięte z ontologii. W celu poprawy przejrzystości, na rys. 6.3 oraz 6.4 ukryto byty obu ontologii.

Obie testowe ontologie dostępne są odpowiednio pod następującymi adresami: `http://oaei.ontologymatching.org/2006/benchmarks/103/onto.rdf` oraz `http://oaei.ontologymatching.org/2006/benchmarks/104/onto.rdf`.

Wynik integracji ontologii wyjściowej z każdą z uogólnionych ontologii był identyczny jak w przypadku złączenia ontologii wejściowej z tożsamą. W obu przypadkach każde z pojęć zostało połączone z odpowiadającym mu pojęciem z uproszczonej ontologii na podstawie podobieństwa $P = 1.0$. Podobnie jak w przypadku pierwszego testu, uzyskano dodatkowe powiązanie pomiędzy klasami „Adres" (ang. *Address*) oraz Referencja (ang. *Reference*) wynikające z niejednoznaczności leksykalnej ich znaczeń.

6.2.5 Eliminacja etykiet pojęć

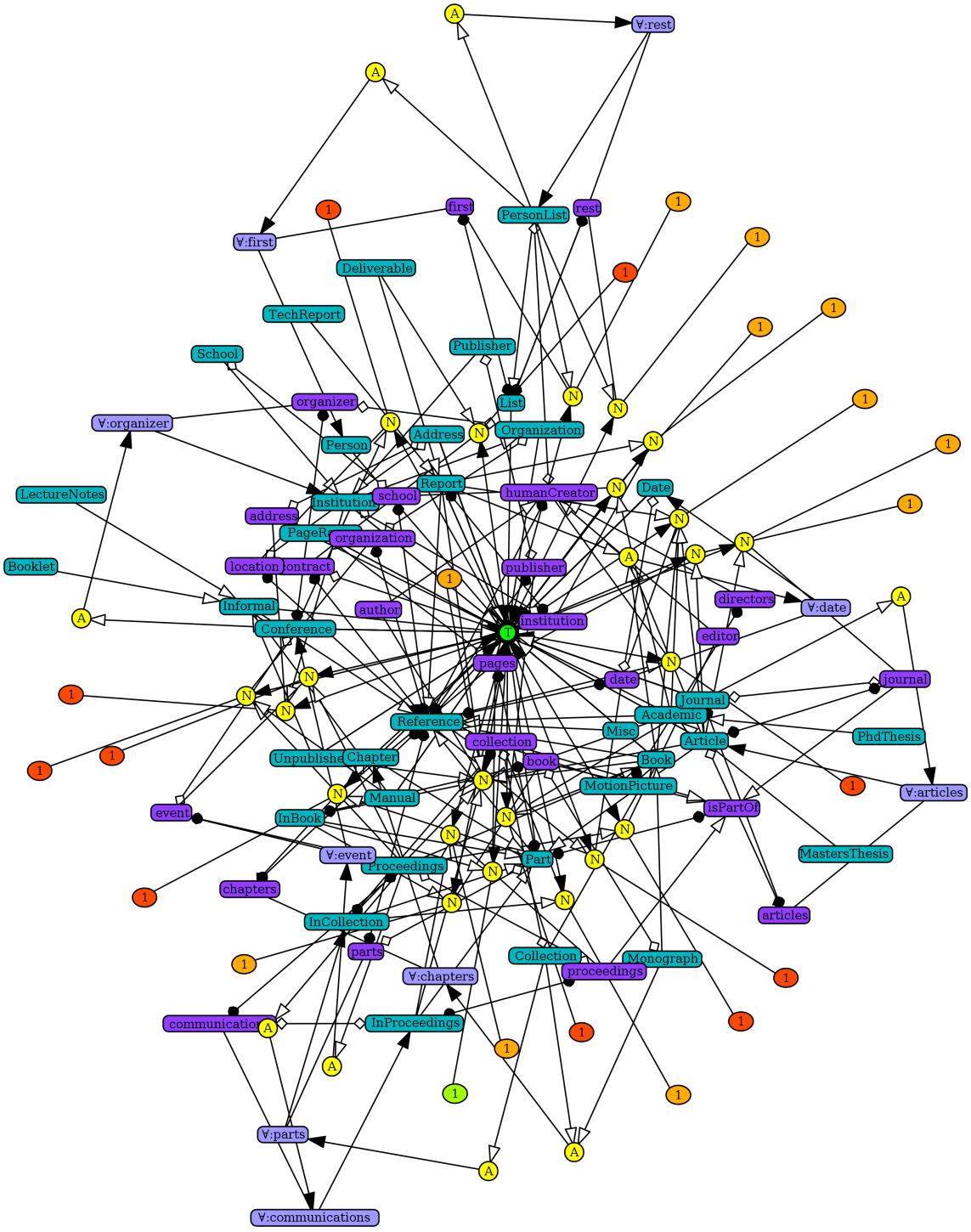
W tym przypadku ontologię wyjściową połączono z identyczną, dostępną pod adresem `http://oaei.ontologymatching.org/2006/benchmarks/201/onto.rdf`, jednak posiadającą etykiety pojęć zastąpione losowymi ciągami znaków. Oryginalne pozostały jedynie trzy pojęcia: *List*, *Organization* oraz *Person*, które autorzy ontologii zaczerpnęli z zewnętrznych źródeł (RDFS [89] oraz FOAF [31]).

Algorytm swoją pracę opierał w tym przypadku na analizie strukturalnej ontologii oraz porównaniu komentarzy przypisanych do pojęć. Wskazówką dla dalszego postępowania były również koncepty o niezmienionych etykietach.

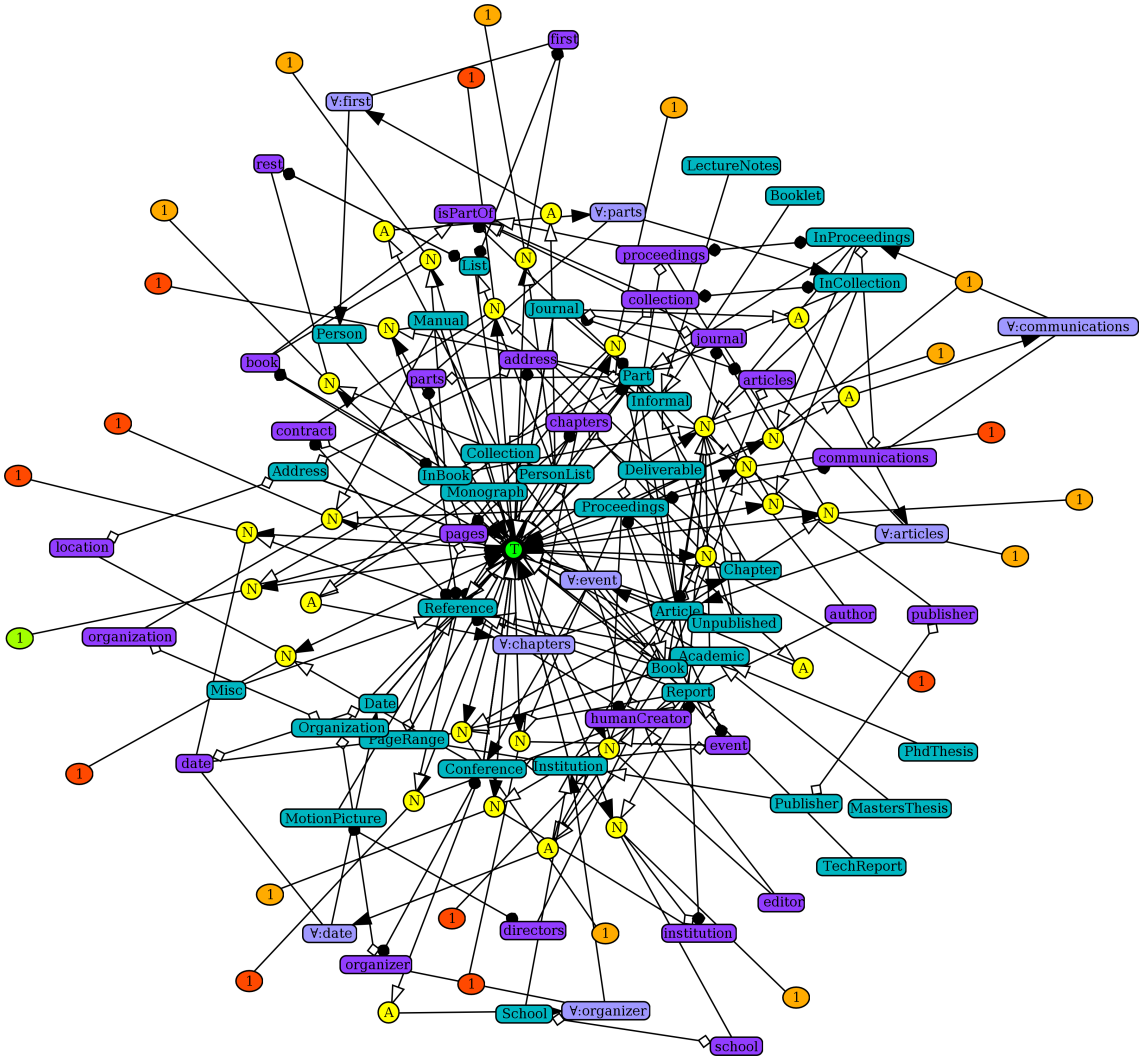
W opisywanym przypadku algorytm pomylił się jedynie w przypadku wiązania pojęć *InCollection* i *Chapter* z ich odpowiednikami ze zmienionymi nazwami. Koncepty *InCollection* i *Chapter* są podklasami tego samego pojęcia *Part*, osadzone są w identycznej strukturze (mają po jednym komentarzu, jednej etykiecie, nie posiadają klas potomnych oraz powiązane są z jedną, tą samą klasą nadrzędną) oraz posiadają zbliżone komentarze (odpowiednio „A part of a book having its own title." oraz „A chapter (or section or whatever) of a book having its own title."). Odpowiadające im koncepty z drugiej ontologii to odpowiednio *dcsqdcsqd* oraz *dzqndbzq*. Wyniki porównań opisanych pojęć zaprezentowano w tablicy 6.1. W wyniku dużego podobieństwa i niejednoznaczności algorytm połączył więc zarówno koncept *InCollection* jak i *Chapter* z pojęciami *dcsqdcsqd* oraz *dzqndbzq*. W wynikowej ontologii wszystkie te pojęcia pośrednio zostały więc uznane za tożsame.

Tablica 6.1: Wyniki wzajemnych porównań pojęć *InCollection* i *Chapter* z pojęciami *dcsqdcsqd* oraz *dzqndbzq*

Lemma A	Lemma B	$max(P_{lex}, P_{sem})$	P_{kom}	P_{str}	P_{sk}	Wynik
InCollection	dcsqdcsqd	0,10	1,00	1,00	1,00	Tożsame
InCollection	dzqndbzq	0,00	0,75	0,67	0,73	Tożsame
Chapter	dcsqdcsqd	0,11	0,75	0,67	0,73	Tożsame
Chapter	dzqndbzq	0,00	1,00	1,00	1,00	Tożsame



Rysunek 6.3: Ontologia reprezentująca atrybuty bibliografii opartej na BibTeX w dialekcie OWL Lite z uogólnionymi relacjami



Rysunek 6.4: Ontologia reprezentująca atrybuty bibliografii opartej na BibTeX w dialekcie OWL Lite z odrzuconymi relacjami

6.3 Konstrukcja ontologii bezpieczeństwa

Działanie algorytmu sprawdzono w trakcie procesu konstrukcji ontologii, będącego właściwym zastosowaniem dla proponowanego rozwiązania. Ontologię bezpieczeństwa [25] skonstruowano zarówno ręcznie, jak i półautomatycznie, korzystając z zaproponowanego algorytmu. Jako bazę dla procesu jej wytwarzania użyto metodologii Noy i McGuinessa [112], wzbogaconej o elementy metodologii NeOn [107, 135, 136] i UPON [44].

Konstrukcję ontologii podzielono na następujące etapy:

1. Analiza wymagań stawianych ontologii (Sekcja 6.3.1),
2. Analiza pojęć bazowych (Sekcja 6.3.2),
3. Konstrukcja modelu bazowego (Sekcja 6.3.3),
4. Implementacja ontologii (Sekcja 6.3.4).

Zaproponowany algorytm użyty został na ostatnim etapie prac w celu sprawdzenia jego efektywności w procesie integracji ontologii.

6.3.1 Analiza wymagań stawianych ontologii

Specyfikację Wymagań Systemowych Ontologii (ang. *ORSD – Ontology Requirement Specification Document*) powstała w oparciu o szablon zaproponowany przez Suarez i in. [134] w ramach projektu NeOn uzupełniony o elementy zaczerpnięte z pracy De Nicoli [44] oraz z drugiej edycji książki Handbook on Ontologies [137]. Utworzony dokument precyzuje wymagania niefunkcjonalne nałożone na ontologię i ma za zadanie odpowiedzieć na następujące pytania (zaczerpnięte z prac Vrandecic'a [137]):

- w jakim celu jest tworzona wspomniana ontologia,
- ustalenie jej zakresu i granic,
- określenie jej przewidywanych zastosowań,
- wyspecyfikowanie wymagań, jakie powinna spełniać tworzona ontologia.

Ponadto dokument zawiera:

- listę wymagań technicznych ontologii,
- listę wymagań jakościowych ontologii,
- listę innych wymagań postawionych przez klienta,
- listę źródeł informacji na temat dziedziny tworzonej ontologii,
- listę innych, możliwych do wykorzystania, ontologii,
- listę wymagań funkcjonalnych, stawianych wobec tworzonej ontologii.

Właściwą treść dokumentu zawarto w kolejnych częściach tego rozdziału.

Cel tworzonej ontologii

Celem konstruowanej ontologii jest stworzenie jednoznacznego modelu semantycznego pojęć z dziedziny bezpieczeństwa. Tak utworzona ontologia powinna być modyfikowalna i łatwa do wykorzystanie w ramach projektów związanych z portalem OCS.

Zakres i granice ontologii

Przyjęto, że tworzona ontologia powinna zawierać pojęcia i zagadnienia z dziedziny bezpieczeństwa (ang. *safety* oraz *security*) w znaczeniu ogólnym, poszerzone o dziedziny blisko związane z bezpieczeństwem, jednak ujęte w ograniczonym zakresie. W szczególności ontologia powinna zawierać:

1. Podstawowe, ogólne terminy z dziedziny bezpieczeństwa.
2. Terminologię z dziedziny bezpieczeństwa informacji (zarówno w ujęciu rozumienia angielskich słów *safety* jak i *security*).
3. Ważniejsze terminy z innych dziedzin bezpieczeństwa, takich jak.:
   (a) drogowego,
   (b) narodowego i międzynarodowego,
   (c) energetycznego.

Powyższe sformułowania definiują zakres tworzonej ontologii dość ogólnie. W związku z tym zdecydowano, że ontologia będzie zawierać terminologię ogólną z dziedziny bezpieczeństwa oraz szczegółową z dziedzin bezpieczeństwa i niezawodności systemów komputerowych oraz wymogów bezpieczeństwa. Za podstawowe źródła wiedzy dla wytwarzanej ontologii przyjęto:

1. Słownik kluczowych terminów bezpieczeństwa informacji NIST [88],
2. Słownik Zarządzania Ryzykiem ENISA [49, 50],
3. Fragment książki „Software Engineering" Iana Sommervilla [128].

Wybór powyższych trzech źródeł wiedzy pozwoli na ukazania zagadnienia bezpieczeństwa z odmiennych punktów widzenia czyniąc ontologię bardziej uniwersalną. Pierwsza z wymienionych publikacji prezentuje bezpieczeństwo informacji z punktu widzenia amerykańskiej jednostki rządowej, druga odnosi się do europejskich i międzynarodowych standardów, a trzecia prezentuje podejście do bezpieczeństwa w aspekcie inżynierii oprogramowania.

Ponadto zdecydowano również o włączeniu do konstruowanej ontologii zagadnień z taksonomii bezpieczeństwa komputerów opracowanej przez IEEE [7] oraz taksonomii wymogów bezpieczeństwa wg Firesmitha [58, 59]. Obie te publikacje cechują się wysokim poziomem formalizacji, co dodatkowo upraszcza konstrukcje na ich podstawie ontologii.

Docelowi użytkownicy

Proponowanych docelowych użytkowników wraz z ich identyfikatorami zaprezentowano w tablicy 6.2.

Przewidywane zastosowania ontologii

Przewidywane zastosowania ontologii wraz z ich identyfikatorami zaprezentowano w tablicy 6.3.

Stosowane metodologie zalecają ponadto, by tworzona ontologia dziedzinowa była tzw. ontologią ogólnego przeznaczenia (ang. *general purpose ontology*).

Tablica 6.2: Docelowi użytkownicy tworzonej ontologii

Id	Opis
U1	Naukowcy i studenci zainteresowani zagadnieniami bezpieczeństwa, ontologiami lub samą ontologią bezpieczeństwa.
U2	Twórcy oprogramowania, którzy chcą wykorzystać ontologię bezpieczeństwa do tworzenia adnotacji semantycznych swoich serwisów sieciowych lub innego oprogramowania.
U3	Programy agentowe lub systemy wyszukujące bądź przetwarzające dane, które zostały adnotowane przy użyciu ontologii bezpieczeństwa.

Tablica 6.3: Zastosowania tworzonej ontologii

Id	Opis zastosowania
Z1	Cele naukowe, w tym testowanie i rozwój portalu OCS.
Z2	Tworzenie serwisu ułatwiającego studentom naukę zagadnień związanych z bezpieczeństwem.
Z3	Aplikacje tworzone z myślą o programowaniu wszechobecnym (ang. *ubiquitous programming*).
Z4	Wyszukiwarki i programy agentowe.

Wymagania niefunkcjonalne

Przewidywane wymagania niefunkcjonalne stawiane wobec tworzonej ontologii wraz z ich identyfikatorami zaprezentowano w tablicy 6.4.

Tablica 6.4: Wymagania niefunkcjonalne tworzonej ontologii

Id	Opis wymagania
WNF1	Ontologia powinna być łatwo modyfikowalna by wspierać język polski.
WNF2	Ontologia powinna wspierać język angielski.
WNF3	Stosowane definicje konceptów i własności powinny pochodzić z uznanych źródeł i standardów.
WNF4	Źródła, z których pochodzą definicje konceptów i własności muszą być podane.
WNF5	Ontologia powinna być rozszerzalna.
WNF6	Ontologia powinna być modyfikowalna.
WNF7	Koncepty i własności powinny być nazwane i opisane w taki sposób, by umożliwić ich łatwe zrozumienie przez człowieka.
WNF8	Ontologia powinna być spójna.
WNF9	Ontologia powinna być kompletna.
WNF10	Ontologia powinna być przenośna.
WNF11	Ontologia powinna działać w systemie OCS.
WNF12	Operacje klasyfikacji na tworzonej ontologii powinny wykonywać się w czasie nie dłuższym niż 1 sekunda.

Wymagania funkcjonalne

Zgodnie z zaleceniami Suarez i in. [134], wymagania funkcjonalne przedstawiono w formie pytań CQ (ang. *competency question*) (Tablica 6.5).

Tablica 6.5: Wymagania funkcjonalne tworzonej ontologii

Id	Pytanie	Przykładowa odpowiedź
CQ1	Czym jest ryzyko?	Prawdopodobieństwem straty.
CQ2	Jakie są rodzaje ataków na systemy komputerowe?	DoS, nieautoryzowany dostęp, itp.
CQ3	Czym jest bezpieczeństwo wewnętrzne?	Stanem bez zagrożeń z wewnątrz.
CQ4	Jakie są atrybuty bezpieczeństwa zewnętrznego?	Dostępność, integralność, poufność.
CQ5	Czym jest atak?	Gwałtowne użycie siły wobec kogoś.

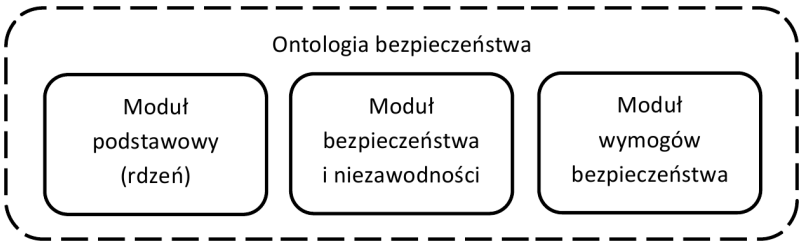
Język implementacji i przenośność ontologii

Konstruowana ontologia powinna zostać zapisana w języku, który umożliwi jej przyszłe zastosowanie przez szeroką gamę aplikacji, w tym system OCS. System OCS wspiera języki obsługiwane przez bibliotekę OWL API [13, 73] w wersji 3.2.3. Należą do nich między innymi RDF, RDFS, OWL 1.1 oraz OWL 2.0. W momencie tworzenia ontologii system OCS był w trakcie przebudowy umożliwiającej wsparcie języka OWL 2.0, stąd zdecydowano się na wybór OWL 1.1 w jego najpopularniejszym wariancie – wersji DL. Jako reprezentację plikową wybrano standard rdf/owl, co pozwoli zachować przenośność ontologii (co wynika z wymagania niefunkcjonalnego WNF10 6.4). Ponadto, kierując się zaleceniami Web Ontology Working Group [142], zdecydowano o braku użycia instancji klas (ang. *individual*). Zamiast nich będą stosowane klasy. Zwiększa to przenośność i łatwość interpretacji modeli ontologicznych.

Podział ontologii

Ontologię docelową podzielono na trzy moduły, w oparciu o tzw. architekturę modularną ontologii proponowaną w ramach Ontology Design Patterns (ODP), opracowaną w ramach projektu NeOn [115] (rys. 6.5). Wyróżnione moduły to:

- moduł podstawowy (rdzeń) ontologii,
- moduł bezpieczeństwa i niezawodności,
- moduł wymogów bezpieczeństwa.



Rysunek 6.5: Moduły ontologii bezpieczeństwa

Język, lokalizacja oraz konwencja nazw w ontologii

Etykiety elementów ontologii opisane zostały w języku angielskim, co pozwoliło bezpośrednio spełnić wymaganie WNF2. Język OWL pozwala na dodawanie etykiet, adnotacji i komentarzy w wielu

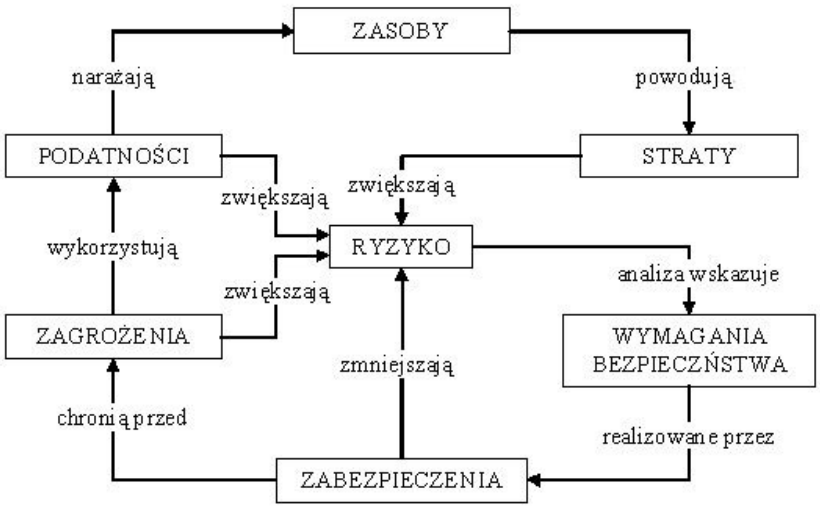
językach jednocześnie, opatrując je międzynarodowym skrótem języka (np. „pl" dla języka polskiego), co umożliwia łatwe tłumaczenie pojęć na inne języki spełniając wymaganie WNF1. Przełączanie pomiędzy dostępnymi językami wykonywane jest bezpośrednio w narzędziu korzystającym z ontologii i nie ma wpływu na wnioskowanie czy inne operacje wykonywane na ontologiach.

Na potrzeby tworzonej ontologii zaadaptowano konwencję nazw zaproponowaną przez Schobera i in. [125] Zaleca ona stosowanie bezkontekstowych i zrozumiałych dla człowieka etykiet (wymaganie WNF7) oraz zabrania stosowania zaprzeczeń, co dodatkowo poprawia czytelność ontologii. Ponadto przyjęto znaną z języka Java systematykę nazywania klas i bytów, czyli tzw. „notację wielbłądzią" (ang. *camel case*). Przyjęto, że nazwy klas i bytów rozpoczynać się będą od dużej litery, a nazwy właściwości od małej. Kolejne słowa nazw wieloczłonowych pisane będą zawsze od dużej litery. Przykładowa nazwa klasy w tej notacji to „PrzykładowaNazwaKlasy" a właściwości „przykładowaNazwaWłaściwości".

6.3.2 Analiza pojęć bazowych

Lista pojęć bazowych, wokoło których konstruowana będzie wynikowa ontologia, opracowana została na podstawie analizy dostępnych źródeł i opracowań zagadnień związanych z bezpieczeństwem. Wzięto pod uwagę trzy grupy opracowań:

- Normy i standardy opisujące zagadnienia bezpieczeństwa – norma ISO 13335-1 oraz słownik IAEA,
- Istniejące ontologie opisujące różne aspekty bezpieczeństwa – ontologia Fenza i Herzoga,
- Literaturę informatyczną opisującą zagadnienie bezpieczeństwa z różnych punktów widzenia.



Rysunek 6.6: Powiązania między pojęciami bazowymi normy ISO 13335-1 [119]

Norma ISO 13335-1

Norma ISO 13335-1 [80], obowiązująca w Polsce na podstawie Polskiej Normy PN-I-13335-1, opisuje pojęcia związane z zarządzaniem ryzykiem w bezpieczeństwie teleinformatycznym. Podstawowe

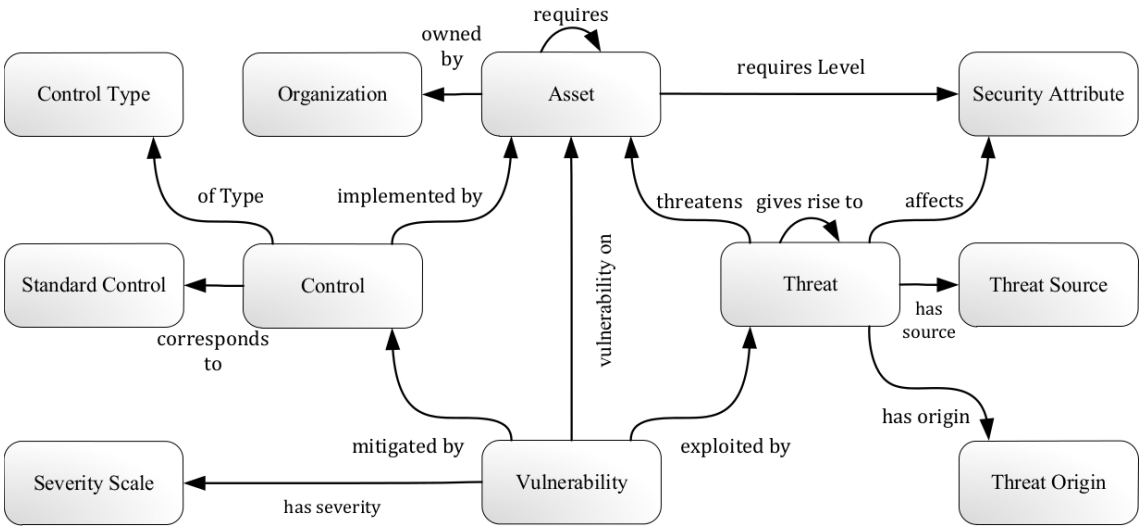
pojęcia ujęte w tej normie to: ryzyko, zasoby, straty, podatności, zagrożenia, zabezpieczenia, wymagania bezpieczeństwa, straty, a ich wzajemne zależności przedstawione zostały na rys. 6.6.

Glosariusz IAEA

IAEA (Międzynarodowa Agencja Energii Atomowej, ang. *International Atomic Energy Agency*) opracowała glosariusz [78] wiążący bezpieczeństwo bezpośrednio z pojęciami: zabezpieczenie, ryzyko, zagrożenie, środki nadzoru.

Ontologia Fenza

Ontologia Fenza [56] jest rozbudowaną ontologią opierającą się na następujących pojęciach bazowych: zabezpieczenie, typ zabezpieczenia, standard zabezpieczeń, aktywa organizacji, organizacja, atrybut bezpieczeństwa, zagrożenie, źródło zagrożenia, przyczyna zagrożenia, podatność. Pojęcia te zostały pozyskane m.in. na podstawie analizy literatury, w tym uznanym wstępie do bezpieczeństwa komputerowego opracowanym przez Guttmana i Robacka [71].



Rysunek 6.7: Podstawowe pojęcia ontologii bezpieczeństwa wg Fenza [56]

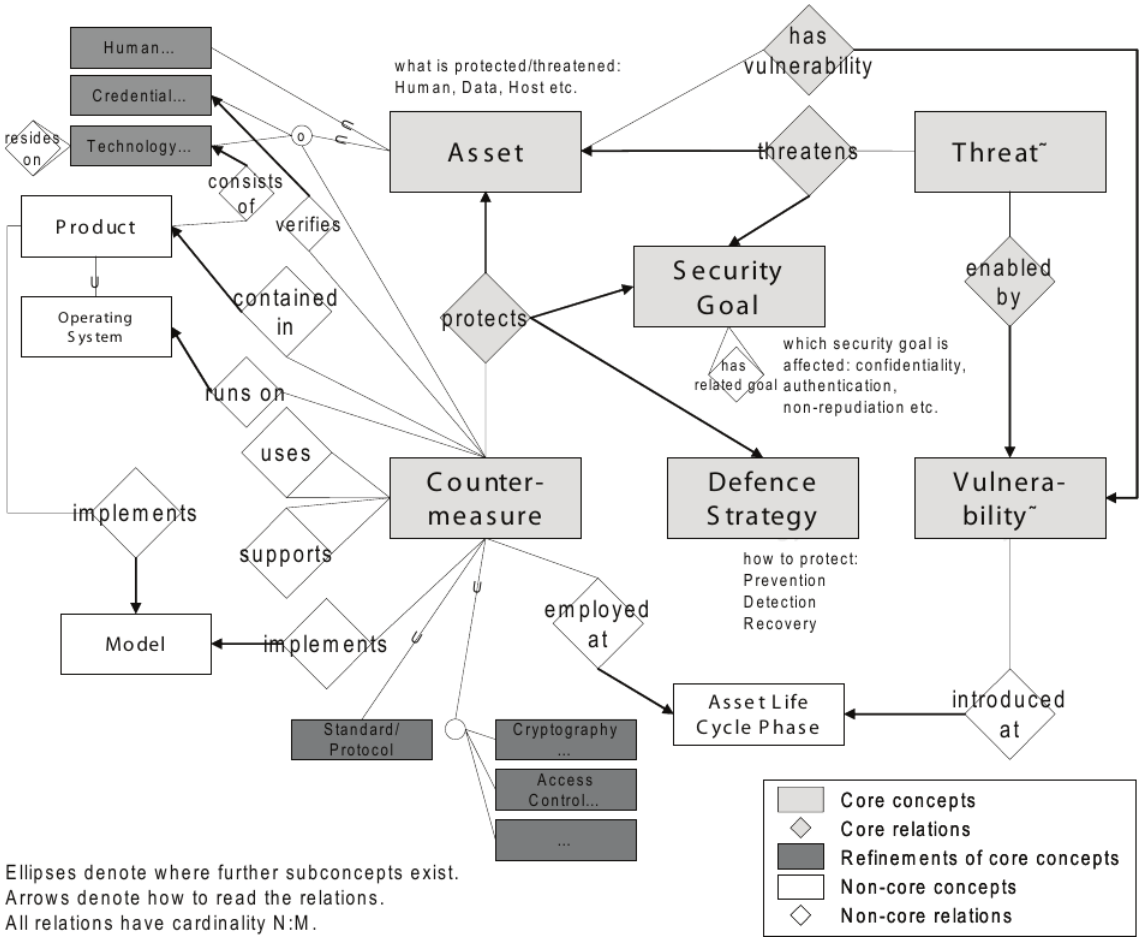
Na rys. 6.7 przedstawiono podstawowe pojęcia tej ontologii wraz z zależnościami pomiędzy nimi. Można tu zauważyć podobieństwo do modelu zarządzania ryzykiem wg standardu ISO 13335-1.

Ontologia Herzoga

Ontologia bezpieczeństwa wg Herzoga [72] została została opracowana głównie w oparciu o książkę Schumachera pt. „Security engineering with patterns: origins, theoretical model, and new applications" [126] oraz ontologię bezpieczeństwa wg Kim [87]. Na rdzeń ontologii składają się następujące pojęcia: aktywa organizacji, środek zaradczy, strategia obrony, cel bezpieczeństwa, zagrożenie, podatność. Jej ogólny zarys przedstawiono na rys. 6.8.

Pojęcia związane z bezpieczeństwem w literaturze

Zagadnienie bezpieczeństwa opisywane jest przez różnych autorów z odmiennego punktu widzenia. Anderson [2] koncentruje się na celach i atrybutach bezpieczeństwa uwzględniając w swoim opracowaniu takie pojęcia jak poufność, integralność i dostępność. Jako opis samego bezpieczeństwa



Rysunek 6.8: Szkielet ontologii Herzoga [72]

używa pojęć: system, podmiot, uczestnik, tożsamość, zaufany, godny zaufania, prywatność, tajemnica, anonimowość, autentyczność, podatność, zabezpieczenie, profil zabezpieczeń, zagrożenie, złamanie bezpieczeństwa. Pojęcia system, podmiot i uczestnik wiążą się z aktywami organizacji. Inne pojęcia mają swoje odniesienia zarówno w normach jak i ontologiach Fenza i Herzoga. Są to podatność, zagrożenie i zabezpieczenie. Pozostałe spośród wymienionych pojęć są w ontologiach Fenza, Herzoga albo uznane za atrybuty bezpieczeństwa (poufność, integralność, dostępność), albo zostały w tych źródłach pominięte.

Sommerville [128] z kolei jako główne pojęcia związane z bezpieczeństwem wewnętrznym uważa: wypadek, groźbę, zagrożenie, szkodę, wagę zagrożenia, prawdopodobieństwo zagrożenia oraz ryzyko. Z bezpieczeństwem zewnętrznym z kolei wiąże pojęcia: odsłonięcie, podatność, groźba i nadzór (w znaczeniu zabezpieczenie). Dodatkowo bezpieczeństwo łączy z niezawodnością, dostępnością i wiarygodnością.

Słownik pojęć bazowych

W tablicy 6.6 porównano występowanie wybranych pojęć w analizowanych źródłach. W zestawieniu zaprezentowano tylko te pojęcia, które występują przynajmniej w dwu źródłach. Analizowane pojęcia zostały w miarę możliwości uogólnione, stąd np. dostępność, wiarygodność, poufność i integralność uznano za atrybuty bezpieczeństwa a nie pojęcia bazowe. System oraz uczestników zakwalifikowano natomiast do aktywów, a nadzór uznano za formę zabezpieczenia.

Tablica 6.6: Zestawienie występowania pojęć bazowych w różnych publikacjach

Pojęcie	ISO	IAEA	Fenz	Herzog	Anderson	Sommerville
Atrybuty (cele) bezpieczeństwa			X	X	X	X
Podatność	X		X	X		X
Ryzyko	X	X				
Szkoda (strata)	X					X
Środek zaradczy (zabezpieczenie, nadzór)	X	X	X	X		X
Zagrożenie, groźba	X	X		X		X
Przedmiot ochrony (aktywa, majątek)	X		X	X	X	

Na podstawie powyższej analizy, jako elementy wstępnej listy pojęć bazowych wybrano:

- atak (ang. *attack*),
- zagrożenie (ang. *threat*),
- ochrona (ang. *protection*),
- bezpieczeństwo (ang. *security*, *safety*),
- zabezpieczenie (ang. *safeguard*),
- ryzyko (ang. *risk*),
- aktywa (ang. *asset*),
- szkoda (ang. *harm*),
- podatność (ang. *vulnerability*),
- groźba (ang. *threat*),
- atrybut bezpieczeństwa (ang. *security feature*),
- dostępność (ang. *availability*),
- integralność (ang. *integrity*),
- poufność (ang. *confidentiality*).

6.3.3 Konstrukcja modelu bazowego

Kolejnym etapem konstrukcji ontologii jest jej formalny model. Teoretycznie możliwe jest zamodelowanie ontologii w postaci zdań języka naturalnego. Przyjmuje ona wtedy postać ciągu zdań wyrażających zależności pomiędzy jej konceptami. Postać ta jej jednak trudna w analizie komputerowej, stąd inżynieria ontologii stosuje do opisu ontologii modele logiczne, a w szczególności dwa najpopularniejsze:

- logika tzw. ramek (ang. *frame logic*, *F-Logic*),
- logika opisowa (ang. *Description Logic*).

Języki oparte na logice ramek

Logika ramek powstała jako połączenie elementów języków programowania obiektowego, języków zapisu wiedzy w postaci ramek (ang. *Frame-based Knowledge Representation*) oraz rachunku predykatów pierwszego rzędu. Umożliwia stosowanie między innymi dziedziczenia i polimorfizmu. Podstawowe pojęcia wprowadzane przez ten model to:

- klasa – zwana też konceptem,
- relacja – opisuje związek pomiędzy dwoma lub więcej konceptami,
- funkcja – to specjalna relacja, w której występują unikalne elementy,
- aksjomat – zdania, które zawsze są prawdziwe,
- instancja – w ontologii reprezentuje pojedyncze byty, czyli instancje (elementy) danej klasy.

Przykładowymi językami opartymi na tym modelu są FLogic, KIF, OBO (Open Biomedical Ontologies). Model ramek jak i oparte na nim języki straciły na znaczeniu po obraniu przez W3C języka OWL jako standardowego języka Sieci Semantycznej.

Języki oparte na logice opisowej

$\mathcal{AL}$	$C, D \longrightarrow$	A	(koncept atomowy)
		$\top$	(koncept uniwersalny)
		$\bot$	(koncept pusty)
		$\neg A$	(negacja atomowa)
		$C \sqcap D$	(przecięcie)
		$\forall R.C$	(restrykcja na wartość)
		$\exists R.\top$	(słaba restrykcja egzystencjalna)
$\mathcal{U}$		$C \sqcup D$	(suma)
$\mathcal{E}$		$\exists R.C$	(restrykcja egzystencjalna)
$\mathcal{C}$		$\neg C$	(negacja)
$\mathcal{N}$		$\leqslant nR$	(najwyżej)
		$\geqslant nR$	(conajmniej
$\mathcal{O}$		$\{a1, a2, \ldots, an\}$	(jeden z)
$\mathcal{I}$	$P, Q, R \longrightarrow$	R^-	(relacja odwrotna)
$\mathcal{Q}$		$\leqslant nR.C$	(kwalifikowana
		$\geqslant nR.C$	restrykcja ilościowa)
$\mathcal{R}$		$P \circ Q \sqsubseteq R$	(złożone zawieranie relacji)

Rysunek 6.9: Podstawowe pojęcia stosowane w logikach opisowych [86]

Logika opisowa (a właściwie cała grupa logik) jest formalną metodą reprezentacji wiedzy [8]. Języki logik opisowych operują na trzech podstawowych elementach [106]:

- koncept (ang. *concept*) – reprezentuje pewien zbiór osobników o wspólnych cechach,
- osobnik (indywiduum, byt, ang. *individual*) – jest instancją pewnego konceptu, będącego klasą podobnych elementów,
- rola (ang. *role*) – jest to binarna relacja pomiędzy dwoma konceptami.

Korzystając z języka logiki opisowej definiujemy terminologię, na jakiej operuje opisywana domena (tzw. TBox, ang. *Terminology Box)*, definiujemy listę osobników w tej domenie występujących (tzw. ABox, ang. *Assertion Box*) oraz relacji pomiędzy nimi (tzw. RBox, ang. *Relation Box*).

Podstawowe pojęcia stosowane w logikach opisowych prezentuje rys. 6.9. Podzbiór zastosowanych pojęć definiuje nazwę danej logiki opisowej. Ponadto podzbiór $\mathcal{ALC}$ często oznaczany jest skrótem $\mathcal{S}$, a litera $\mathcal{H}$ oznacza możliwość budowy hierarchii ról.

Logiki opisowe zostały zaadaptowane do tworzenia ontologii dzięki możliwości przeprowadzania za ich pomocą wnioskowania. Należy jednak zauważyć, że proces ten jest zazwyczaj wykładniczy.

Na podstawie logiki opisowej opracowano język DAML+OIL, który był pierwszym językiem tego typu zatwierdzonym przez W3C (World Wide Web Consortium) [74]. Na jego bazie, poprzez poszerzenie składni i rozbudowę aparatu pojęciowego opracowano język OWL [45], który w 2004 roku stal się standardem W3C. W 2009 roku wprowadzono jego rozszerzenie opatrzone numerem OWL 2.0.

Obecnie funkcjonują następujące wersje języka OWL:

- OWL 1.0 i OWL 1.1, wraz z dialektami:
  - OWL Lite – uproszczona wersja języka, na logice opisowej $\mathcal{SHIQ}$,
  - OWL DL – domyślna wersja języka, oparta na logice opisowej $\mathcal{SHOIN}$,
  - OWL Full – podobny do DL, dopuszcza cykle w relacjach oraz zezwala na traktowanie własności oraz bytów jak klasy (i vice versa) co zwiększa złożoność obliczeniową wnioskowania nie dając gwarancji zakończenia tego procesu w skończonym czasie.
- OWL 2.0, oparty na logice opisowej $\mathcal{SROIQ}$.

Najpopularniejszym jest obecnie dialekt OWL DL 1.1 [137], lecz jest on sukcesywnie wypierany przez OWL 2.0, które gwarantuje skończoność czasu wnioskowania oraz dostarcza większe możliwości opisowe.

Modelowanie pojęć bazowych

Zastosowanie języka OWL DL, jako formy zapisu tworzonej ontologii, niesie za sobą konieczność zastosowania logiki opisowej $\mathcal{SHOIN}$ do zamodelowania ontologii. Za jej pomocą opisano podstawowe pojęcia tworzonej ontologii.

Bezpieczeństwo Wcześniej w tej pracy bezpieczeństwo określono jako stan, w którym nie występują zagrożenia. Za pomocą logiki opisowej można to zapisać jak w wyrażeniu 6.1

$$\texttt{Bezpieczeństwo} \equiv \texttt{Stan} \sqcap \neg\exists\texttt{występuje.Zagrożenie} \tag{6.1}$$

Ponadto wyróżniamy pojęcia bezpieczeństwa wewnętrznego i zewnętrznego, jednak granica pomiędzy nimi nie jest jasno sprecyzowana, stąd nie ma konieczności tworzenia pomiędzy nimi dodatkowej relacji (wyrażenie 6.2).

$$\begin{aligned} \texttt{BezpieczeństwoWewnętrzne} &\sqsubseteq \texttt{Bezpieczeństwo} \\ \texttt{BezpieczeństwoZewnętrzne} &\sqsubseteq \texttt{Bezpieczeństwo} \end{aligned} \tag{6.2}$$

Atrybuty i cele bezpieczeństwa Bezpieczeństwo opisane jest określonymi atrybutami (wyrażenie 6.3).

$$\exists\texttt{posiadaAtrybut.AtrybutBezpieczeństwa} \sqsubseteq \texttt{Bezpieczeństwo} \tag{6.3}$$

Zestaw atrybutów uzależniony jest od opisywanej dziedziny bezpieczeństwa. Bezpieczeństwo informacyjne opisane w normie ISO 13335-1 definiuje atrybuty: poufność, autentyczność, dostępność, integralność, rozliczalność, niezawodność (wyrażenia 6.4).

$$\begin{aligned} &\texttt{AtrybutBezpieczeństwaInformacyjnegoISO13335} \equiv \{\texttt{Poufność, Autentyczność,} \\ &\qquad\qquad \texttt{Dostępność, Integralność,Rozliczalność, Niezawodność}\} \\ &\texttt{AtrybutBezpieczeństwaInformacyjnegoISO13335} \sqsubseteq \texttt{AtrybutBezpieczeństwa} \end{aligned} \tag{6.4}$$

Podatność Słownik WordNet podatność łączy bezpośrednio z odsłonięciem poprzez stwierdzenie „Odsłonięcie jest stanem, gdy występuje podatność" (wyrażenie 6.5).

$$\texttt{Odsłonięcie} \equiv \texttt{Stan} \sqcap \exists \texttt{występuje.Podatność} \tag{6.5}$$

Podatność z kolei występuje wtedy, gdy podmiot nie posiada odporności na szkodę lub zagrożenie (wyrażenie 6.6).

$$\begin{aligned} \texttt{Podatność} \equiv &\neg\exists \texttt{istniejeOdporność.Szkoda} \\ &\sqcup \neg\exists \texttt{istniejeOdporność.Zagrożenie} \end{aligned} \tag{6.6}$$

Polityka bezpieczeństwa Definicja pojęcia polityki bezpieczeństwa w dużej mierze uzależniona jest od kontekstu jego użycia, a jego znaczenie różne. W interesującej nas dziedzinie wyróżnić możemy politykę bezpieczeństwa organizacji i politykę bezpieczeństwa informacyjnego. Norma ISO 13335-1 definiuje politykę bezpieczeństwa jako „zbiór ogólnych zasad i podstawowych wymagań określających w jaki sposób powinny być zarządzane, udostępniane i chronione przed nieupoważnionym wykorzystaniem, zniszczeniem lub nieautoryzowanymi zmianami materialne i informacyjne aktywa organizacji" [80]. Zdanie to można zapisać jak w wyrażeniu 6.7.

$$\begin{aligned} \exists \texttt{zawiera.}(&\texttt{ZbiórZasadPolitykiBezpieczeństwa} \\ &\sqcup \texttt{ZbiórWymagańPolitykiBezpieczeństwa}) \\ &\sqsubseteq \texttt{PolitykaBezpieczeństwa} \end{aligned} \tag{6.7}$$

Ryzyko Istnieje wiele definicji pojęcia ryzyka, co czyni je dość mało precyzyjnym. W takich przypadkach zalecany jest wybór najczęściej stosowanych definicji i na ich podstawie opracowanie zbioru nierozłącznych podklas definiowanego pojęcia. Na potrzeby konstrukcji ontologii przyjęto, że ryzyko informatyczne jest jednym z rodzajów ryzyka (wyrażenie 6.8).

$$\texttt{RyzykoInformatyczne} \sqsubseteq \texttt{Ryzyko} \tag{6.8}$$

Norma ISO 13335-1 definiuje ryzyko jako prawdopodobieństwo straty (wyrażenie 6.9).

$$\exists \texttt{jestPrawdopodobieństwemWystąpienia.Strata} \sqsubseteq \texttt{Ryzyko} \tag{6.9}$$

Szkoda Wg słownika WordNet szkoda to straty materialne, moralne, krzywda (wyrażenie 6.10).

$$\texttt{Szkoda} \equiv \texttt{StrataMaterialna} \sqcup \texttt{StrataMoralna} \sqcup \texttt{Krzywda} \tag{6.10}$$

Zabezpieczenie W połączeniu z bezpieczeństwem w literaturze można napotkać pojęcia zabezpieczenie, środek zaradczy i nadzór [78]. Wg. słownika WordNet „zabezpieczenie to coś, co chroni". Zabezpieczenie jest to także „środek zaradczy przeciwko zagrożeniu" (wyrażenie 6.11).

$$\begin{aligned}\texttt{Zabezpieczenie} \equiv \exists\texttt{chroni.PrzedmiotOchrony} \sqcup \texttt{ŚrodekZaradczy} \\ \sqcap \exists\texttt{przeciwdziała.Zagrożenie}\end{aligned} \tag{6.11}$$

Środek zaradczy z kolei jest to czynność mająca na celu przeciwdziałanie innej czynności lub zdarzeniu (wyrażenie 6.12).

$$\begin{aligned}\texttt{ŚrodekZaradczy} \equiv \texttt{Czynność} \\ \sqcap (\exists\texttt{przeciwdziała.Czynność} \sqcup \exists\texttt{przeciwdziała.Zdarzenie})\end{aligned} \tag{6.12}$$

Zagrożenie i groźba WordNet definiuje groźbę jako coś będące źródłem zagrożenia (wyrażenie 6.13), a zagrożenie opisuje jako stan występującej podatności na wystąpienie szkody (wyrażenie 6.14).

$$\texttt{Groźba} \equiv \exists\texttt{jestŹródłem.Zagrożenie} \tag{6.13}$$

$$\texttt{Zagrożenie} \equiv \texttt{Stan} \sqcap \exists\texttt{występuje.Podatność} \tag{6.14}$$

Zarówno WordNet jak i ontologia Fenza podają ponadto, że zagrożenie może być źródłem kolejnego zagrożenia (wyrażenie 6.15).

$$\exists\texttt{jestŹródłem.Zagrożenie} \sqsubseteq \texttt{Zagrożenie} \tag{6.15}$$

Przedmioty ochrony Przedmiot ochrony w dużej mierze zależy od dziedziny bezpieczeństwa opisywana ontologią. Norma ISO 13335-1 definiuje aktywa organizacji jako podstawowy przedmiot ochrony w ramach bezpieczeństwa informacji (wyrażenie 6.16).

$$\texttt{AktywaOrganizacji} \sqsubseteq \texttt{PrzedmiotOchrony} \tag{6.16}$$

6.3.4 Implementacja ontologii

Metodologia

Ontologię wytworzono techniką iteracyjno-przyrostową. Moduł podstawowy opracowany został w oparciu na 3 niewielkich (poniżej 100 klas) ontologiach [24]. Ontologie te zostały scalone do postaci pojedynczego pliku OWL. W skład modułu bezpieczeństwa i niezawodności weszła jedna ontologia opracowana na podstawie taksonomii Avižienisa [7]. Moduł wymogów bezpieczeństwa także oparty został na pojedynczej ontologii opartej na taksonomii Firesmitha [58, 59]. Te trzy moduły na ostatnim etapie scalone zostały w jedną, finalną ontologię.

Procedura konstrukcji ontologii, oparta na metodologii Noya i McGuinessa wspartej o zalecenia z metodologii UPON i NeON, składa się z 8 kroków:

1. Tworzenie leksykonu – wybór zakresu terminów z analizowanych glosariuszy i taksonomii, terminy te będą tematyczną bazą tworzonej ontologii.

2. Wybór konceptów – koncepty tworzone są bezpośrednio z leksykonu poprzez wybór wszystkich nazw własnych. Następnie zbiór ten został rozszerzony o nazwy własne wydobyte z definicji wcześniej wybranych pojęć. Zbiór ten został następnie przekształcony na klasy w języku OWL. Klasy te opatrzone zostały komentarzem tożsamym z definicją konceptu, na podstawie którego powstała.

3. Tworzenie hierarchii konceptów – hierarchia pojęć zamodelowana została za pomocą relacji *subClassOf* („podklasa") języka OWL. Utworzona została na podstawie relacji dziedziczenia występującego w taksonomiach („pojęcie A dziedziczy po pojęciu B" jest tożsame z stwierdzeniem „klasa A jest podklasą B") oraz występujących w definicjach konceptów zwrotów „jest rodzajem" (zwrot „A jest rodzajem B" jest tożsame z „klasa A jest podklasą B").

4. Wybór konceptów rozłącznych i synonimów – jako rozłączne oznaczono klasy wzajemnie wykluczające się. Zamodelowano to relacją *disjointWith* języka OWL. Synonimy zamodelowane zostały za pomocą relacji *equal*. Przy tworzeniu tych relacji posiłkowano się słownikiem WordNet oraz doświadczeniem autorów ontologii.

5. Konstrukcja relacji konceptów – relacje konstruowane są na podstawie czasowników łączących wcześniej wybrane pojęcia. Z taksonomii reprezentowanych w postaci graficznej wyłuskano relację agregacji i przekształcono na relację *hasPart* języka OWL.

6. Tworzenie hierarchii relacji – relacje, podobnie jak koncepty, grupowane są na podstawie czasowników występujących w ich definicjach. Czynność ta jest opcjonalna.

7. Precyzowanie relacji – relacje są doprecyzowane poprzez określenie ich dziedziny oraz przeciwdziedziny. Dodatkowo relacje określane są jako funkcjonalne czy przeciwne względem innej relacji.

8. Integracja – etap łączenia utworzonej ontologii z innymi wchodzącymi w skład modułu.

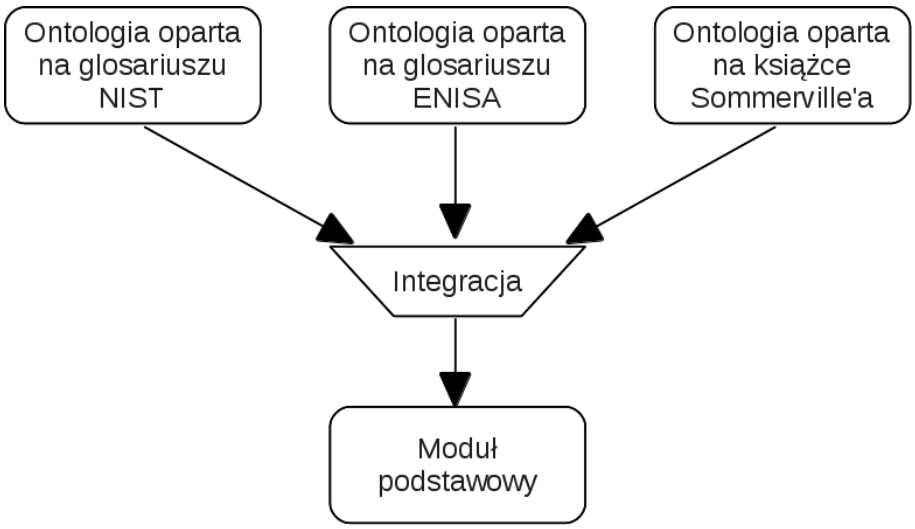
Każda z ontologii wchodzących w skład tworzonej ontologii bezpieczeństwa powstała z wykorzystaniem powyższego schematu.

Moduł bazowy

Moduł bazowy powstał w oparciu na podstawowych pojęciach z dziedziny analizy ryzyka:

- ryzyko (ang. *risk*),
- aktywa (ang. *asset*),
- podatność (ang. *vulnerability*),
- groźba (ang. *threat*),
- zabezpieczenie (ang. *safeguard*).

Zbiór ten następnie rozszerzono o pojęcia zaczerpnięte z definicji tych pojęć oraz pojęcia wyspecyfikowane w sekcji 6.3.2 (o ile występowały w źródłach, na których oparta była ontologia modułu bazowego).



Rysunek 6.10: Konstrukcja modułu podstawowego ontologii bezpieczeństwa

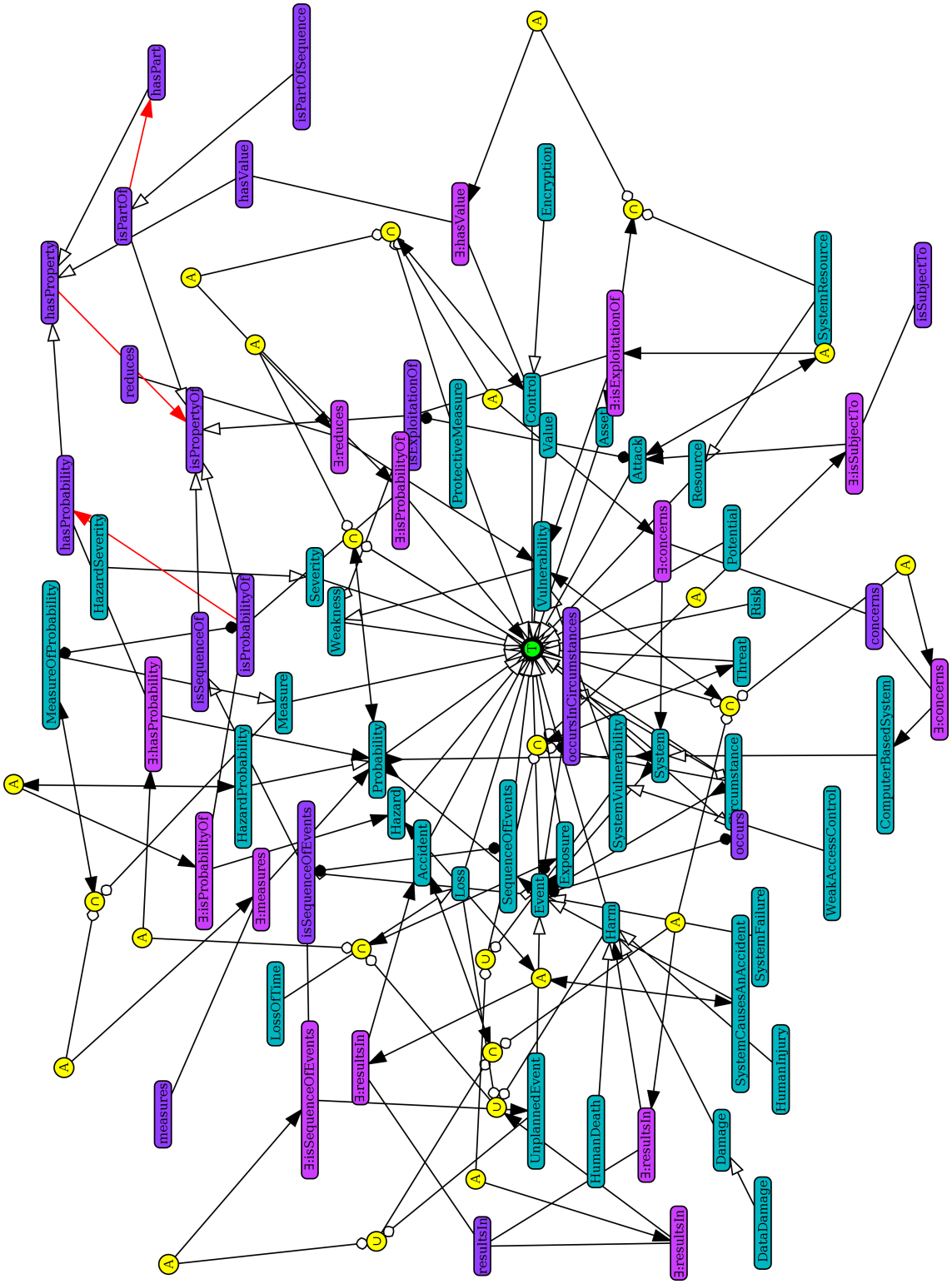
W trakcie prac nad modułem bazowym opracowano trzy ontologie, każda oparta na jednym ze źródeł wyspecyfikowanych w sekcji 6.3.1. Te trzy ontologie zostały następnie scalone w jedną, właściwą ontologię. Proces ten zobrazowano na rys. 6.10.

Każda z wytworzonych ontologii opatrzona została jednoznacznym identyfikatorem URI (ang. *Universal Resource Identiffier*), jednoznacznie definiującym jej przestrzeń nazw, a tym samym eliminującym konflikty z innymi ontologiami.

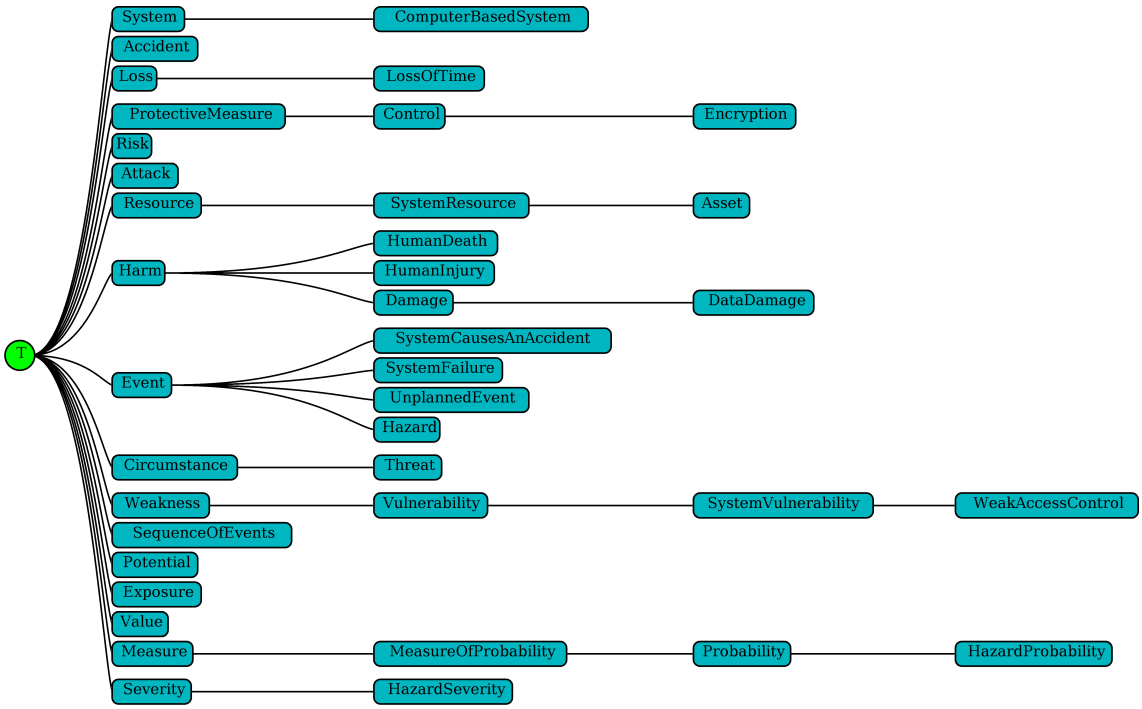
Ontologia oparta na glosariuszu ENISA składa się z 43 klas oraz 28 właściwości, ontologia oparta na glosariuszu NIST z 70 klas i 23 właściwości. Obie te ontologie zostały wcześniej zaprezentowane w rozdziale 4.6 odpowiednio na rys. 4.3 i 4.4 (pełna reprezentacja ontologii) oraz rys. 4.5 i 4.6 (wywnioskowana hierarchia klas). Trzecia z ontologii, oparta na książce Sommerville'a, składa się z 40 klas i 22 właściwości. Jej pełna wizualizacja znajduje się na rys. 6.11, a wywnioskowana hierarchia klas na rys. 6.12.

W celu scalenia ontologii w Moduł bazowy użyto dedykowanych narzędzi, które w założeniu miały wykonać tę czynność półautomatycznie. Dostępne narzędzia niestety okazały się niestabilne, a generowane przez nie ontologie niespójne. Przeprowadzono następujące próby automatycznego i półautomatycznego scalenia ontologii:

- Protégé 4.0.2 z funkcją Merge Ontologies [110, 132] – mechanizm łączy dwie lub więcej ontologii w jedną identyfikując koncepty na podstawie przestrzeni nazw. Koncepty, nawet te o identycznych nazwach, należące do innych przestrzeni nazw zawsze traktowane są jako różne, stąd wynikowa ontologia posiadała jedynie trzy rozłączne drzewa konceptów i właściwości w żaden sposób nie powiązane ze sobą.
- PROMPT [113, 114] – funkcjonalność narzędzia z założenia umożliwia pełną integrację ontologii. Niestety brak rozwoju, a przez to brak wsparcia dla wielu konstrukcji języka OWL, powoduje, że jest to narzędzie bardzo niestabilne. W przypadku rozpatrywanych ontologii, liczne błędy aplikacji uniemożliwiały pracę, czy zapis wyników integracji.
- ContentMap [83] – narzędzie umożliwiające scalanie ontologii na podstawie wcześniej przygotowanego opisu odwzorowań w języku zgodnym z Alignment API [52]. Do wygenerowania pliku odwzorowań wykorzystano narzędzie FalconAO [24, 81]. Niestety uzyskane w ten sposób ontologie okazały się niespójne, przez co niemożliwe stało się wnioskowanie na nich. Ponadto warto zauważyć, że zastosowanie drzewa decyzyjnego w trakcie pracy narzędzia wydłużyło czas integracji z kilku minut do kilku godzin nie zwiększając jakości złączenia.



Rysunek 6.11: Ontologia bezpieczeństwa utworzona na bazie książki Sommerville'a – pełna ontologia



Rysunek 6.12: Ontologia bezpieczeństwa utworzona na bazie książki Sommerville'a – wywnioskowana hierarchia klas

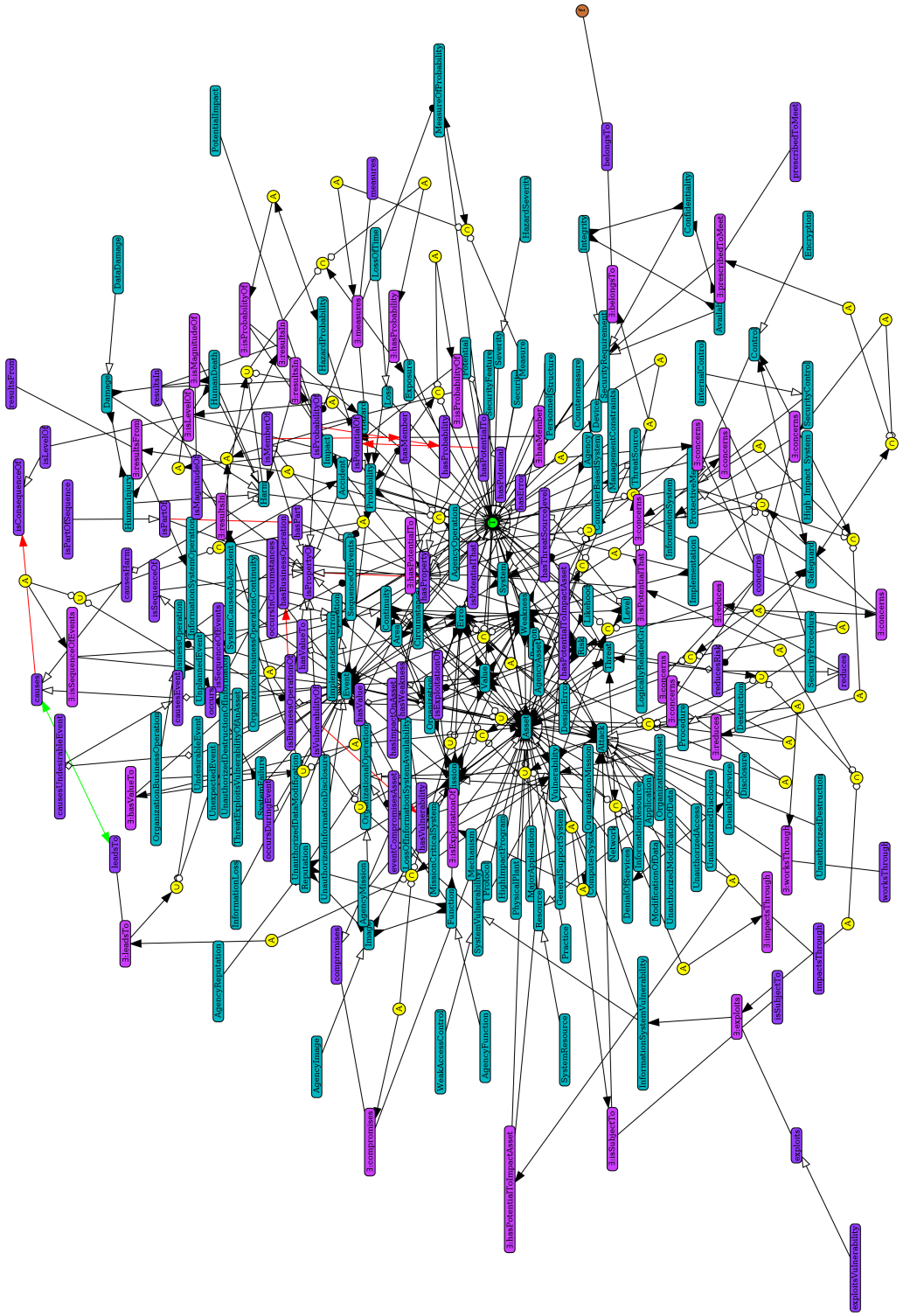
- Ontology Module Composition [42] – narzędzie w swojej funkcjonalności zbliżone do Merge Ontologies edytora Protégé. Posiada dwa tryby pracy: z uwzględnieniem przestrzeni nazw (działa wtedy identycznie jak moduł Protégé) oraz z brakiem różnicowania konceptów poprzez przestrzenie nazw. Niestety wybór drugiej opcji spowodował „znikanie" części klas o identycznych nazwach z ontologi wynikowej oraz przypisanych do nich etykiet, komentarzy i definicji, przez co narzędzie okazało się nieużyteczne.

W wyniku niepowodzenia przy zautomatyzowanym scalaniu ontologii zdecydowano się na realizację tego procesu w sposób półautomatyczny. Ontologie integrowano poprzez porównanie ich narzędziem FalconAO, a następnie ręcznym wprowadzeniu wyników jego pracy do ontologi wynikowej poprzez zastosowanie relacji „subClassOf", „subPropertyOf", „equivalentClass" oraz „equivalentProperty" języka OWL za pośrednictwem edytora Protégé z uwzględnieniem następujących zasad:

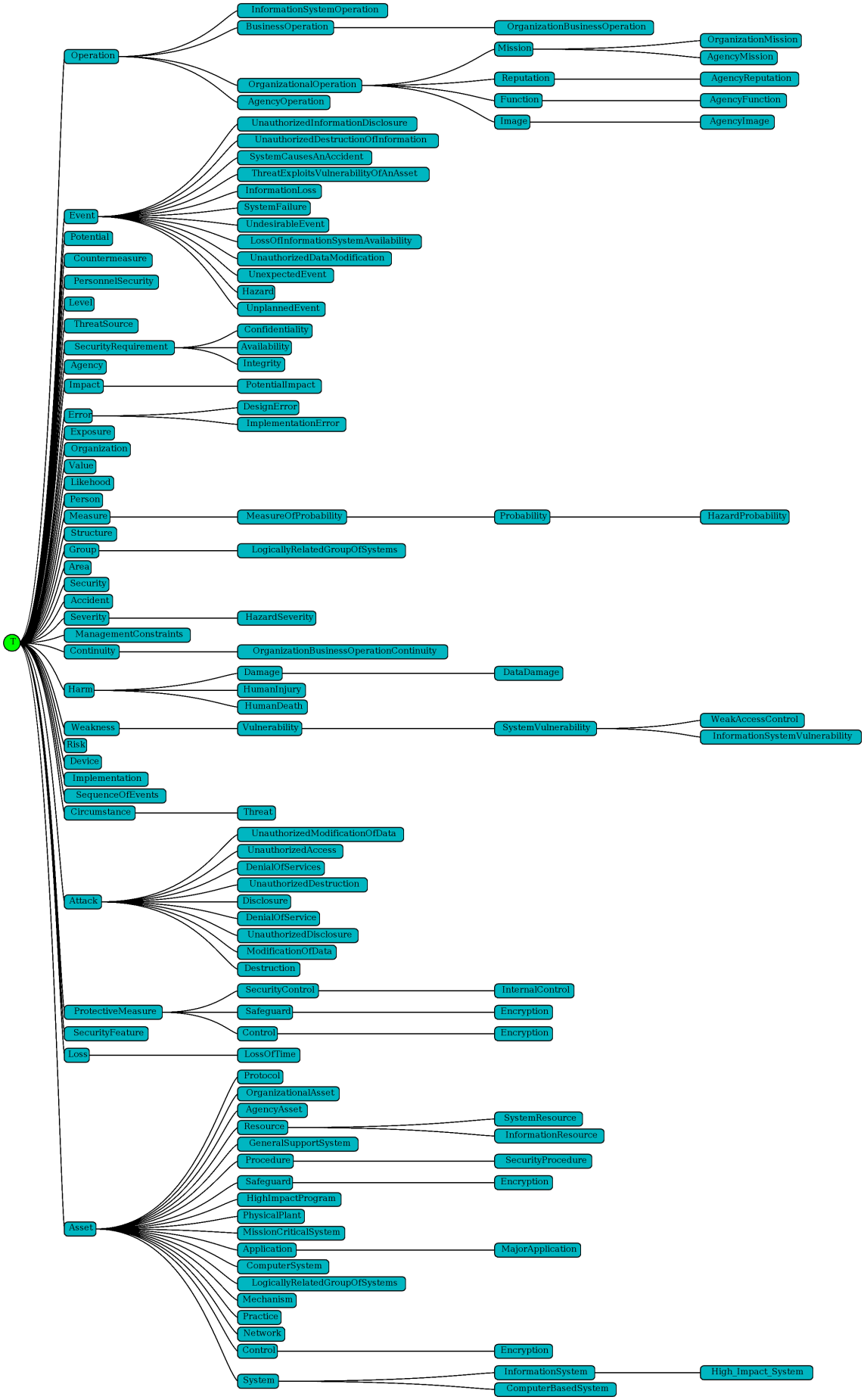
- analizując znaczenie i definicje pojęć oraz właściwości stwierdzano ich wzajemne relacje, jeżeli jedno z pojęć lub właściwości okazywało się szersze lub bardziej ogólne znaczeniowo, stosowano relacje „subClassOf" lub „subPropertyOf",
- analizując znaczenie i definicje pojęć oraz właściwości stwierdzono tożsamość, odpowiadające im elementy obu ontologii były łączone relacją „equivalentClass" lub „equivalentProperty" lub integrowane w jeden przy identyczności nazwy i definicji,
- gdy definicje pojęć znacząco różnią się tworzono dla nich pojęcie nadrzędne i wiązano relacją dziedziczenia.

Po połączeniu ontologii ujednolicono przestrzenie nazw wszystkich łączonych elementów z zachowaniem wszystkich aksjomatów z obu ontologii źródłowych. Po zakończeniu scalania weryfikowano ontologię i uzupełniano ją o odniesienia do źródeł pochodzenia pojęć i ich definicji. Wyni-

kowa scalona ontologia zawiera 122 klasy i 63 właściwości. Jej pełna wizualizacja znajduje się na rys. 6.13, a wywnioskowana hierarchia klas na rys. 6.14.



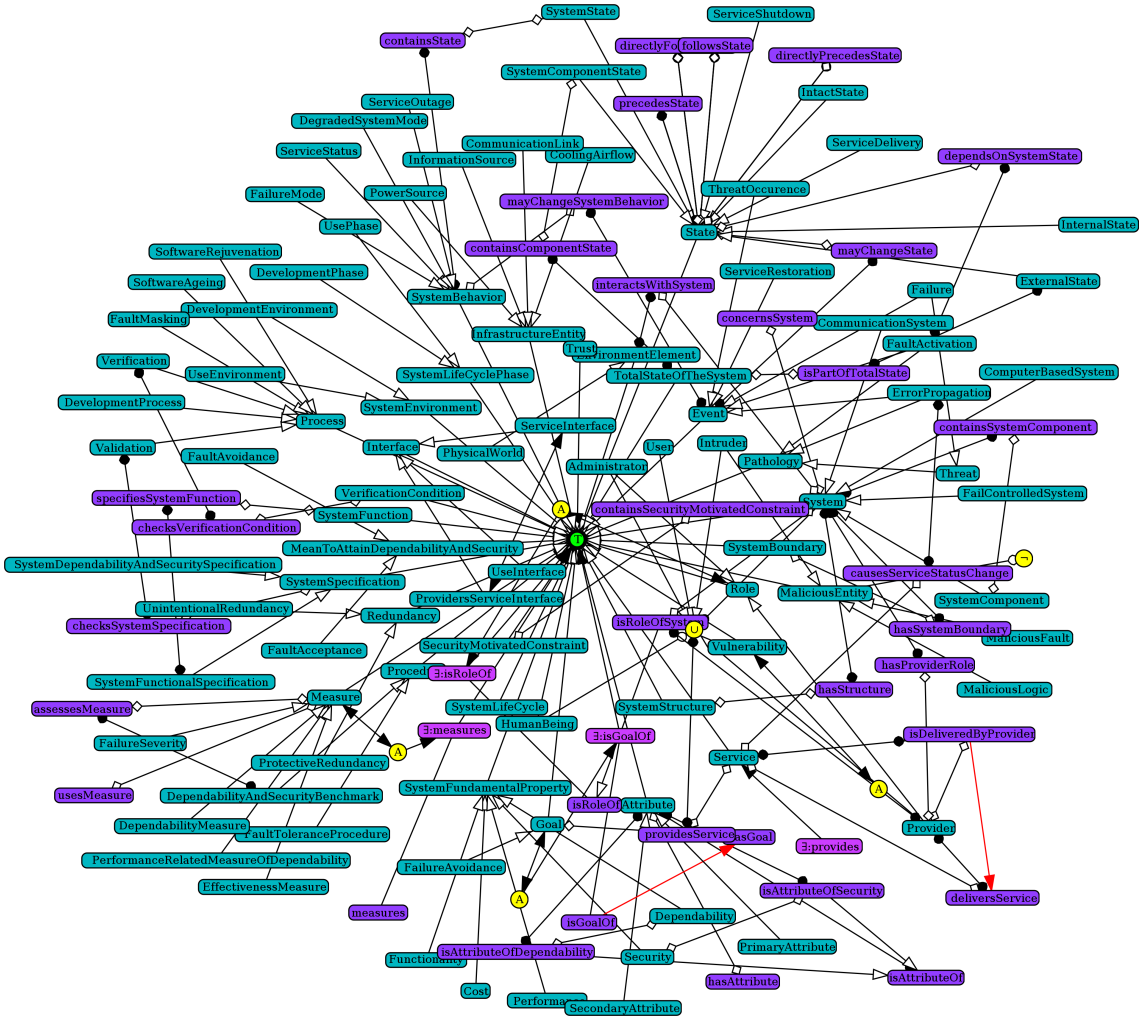
Rysunek 6.13: Moduł bazowy ontologii bezpieczeństwa – pełna ontologia



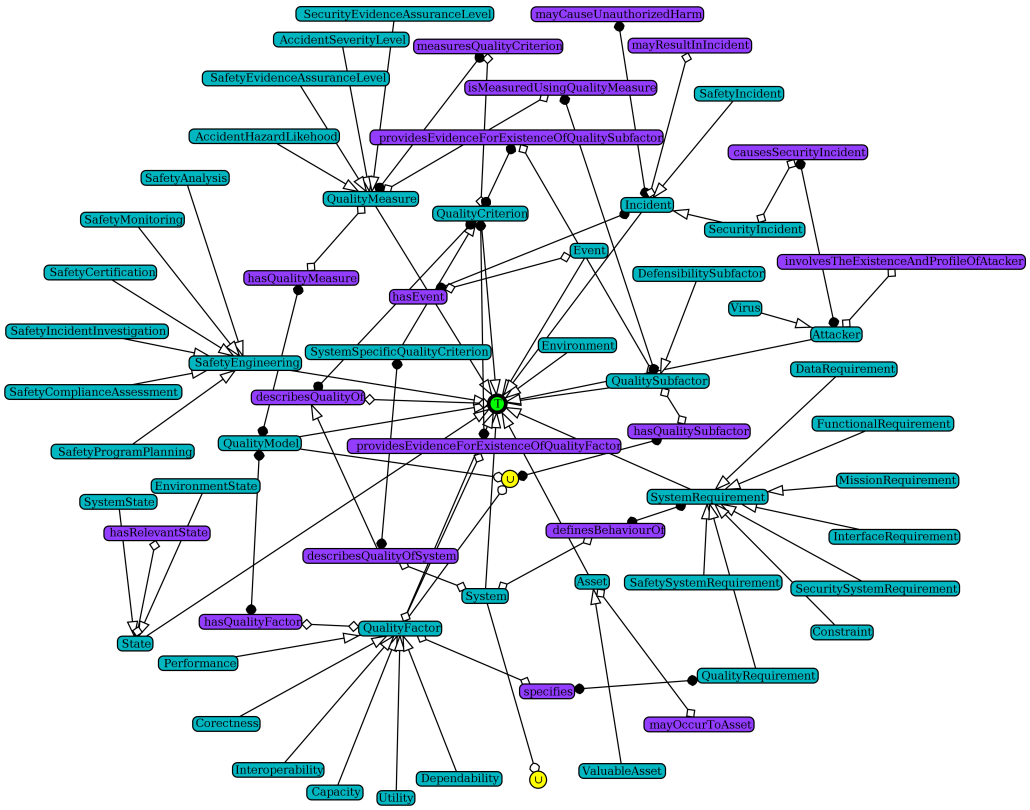
Rysunek 6.14: Moduł bazowy ontologii bezpieczeństwa – wywnioskowana hierarchia klas

Moduł bezpieczeństwa i niezawodności oraz Moduł wymogów bezpieczeństwa

Każdy z pozostałych modułów ontologii bezpieczeństwa składa się z pojedynczej, rozbudowanej ontologii. W skład Modułu bezpieczeństwa i niezawodności weszła jedna ontologia opracowana na podstawie taksonomii Avižienisa [7]. Składa się ona z 269 klas i 91 właściwości. Wizualizacja tej ontologi zaprezentowana jest na rys. 6.15. Z kolei Moduł wymogów bezpieczeństwa oparty został o taksonomię Firesmitha [58, 59]. Składa się ona z 195 klas i 56 właściwości. Wizualizacja tej ontologi zaprezentowana jest na rys. 6.16. W obu przypadkach, ze względu na znaczne rozmiary ontologii, zaprezentowano wizualizację jedynie części elementów. Prezentowane są jedynie te węzły, które oddalone są od węzła konceptu *owl:Thing* o nie więcej niż dwie krawędzie. Pominięto również krawędzie obrazujące rozłączność klas ontologii.



Rysunek 6.15: Moduł bezpieczeństwa i niezawodności ontologii bezpieczeństwa – wizualizacja fragmentu ontologii



Rysunek 6.16: Moduł wymogów bezpieczeństwa ontologii bezpieczeństwa – wizualizacja fragmentu ontologii

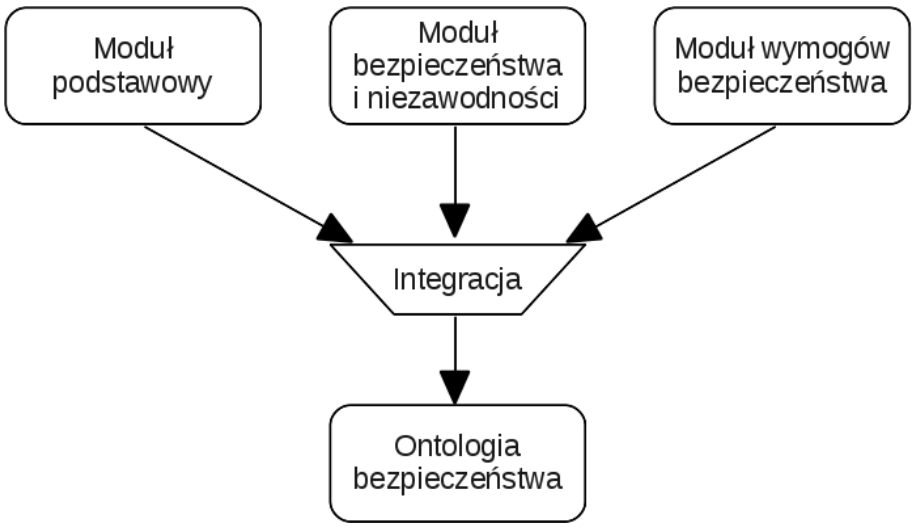
Ontologia bezpieczeństwa

Moduł bazowy, Moduł bezpieczeństwa i niezawodności oraz Moduł wymogów bezpieczeństwa zostały scalone zgodnie z procedurą opisaną w sekcji 6.3.4 tworząc właściwą ontologię bezpieczeństwa (rys. 6.17).

W rezultacie prac powstała pojedyncza ontologia opisująca zagadnienia z dziedziny bezpieczeństwa informatycznego. Jest to duża ontologia składająca się z 566 klas oraz 192 właściwości. Wizualizację jej struktury zaprezentowano na rys. 6.18. Ze względu na znaczne rozmiary ontologii zaprezentowana wizualizacja obrazuje jedynie część elementów. Prezentowane są jedynie klasy, które oddalone są od węzła konceptu *owl:Thing* o nie więcej niż dwie krawędzie. Pominięto również krawędzie obrazujące rozłączność klas ontologii oraz właściwości.

6.3.5 Automatyczna integracja modułów

Kolejne kroki konstrukcji ontologii bezpieczeństwa, opisane w rozdziale 6.3.4, polegające na integracji mniejszych ontologii, przeprowadzono za pomocą zaprezentowanego w niniejszej pracy algorytmu, a następnie wynik jego działania porównano z ontologiami pozyskanymi w procesie ręcznego łączenia ontologii składowych. Integrację przeprowadzono w systemie OCS każdorazowo eksportując zintegrowaną ontologię do osobnego pliku, ujednolicając przy tym URI wynikowej ontologii.



Rysunek 6.17: Konstrukcja ontologii bezpieczeństwa

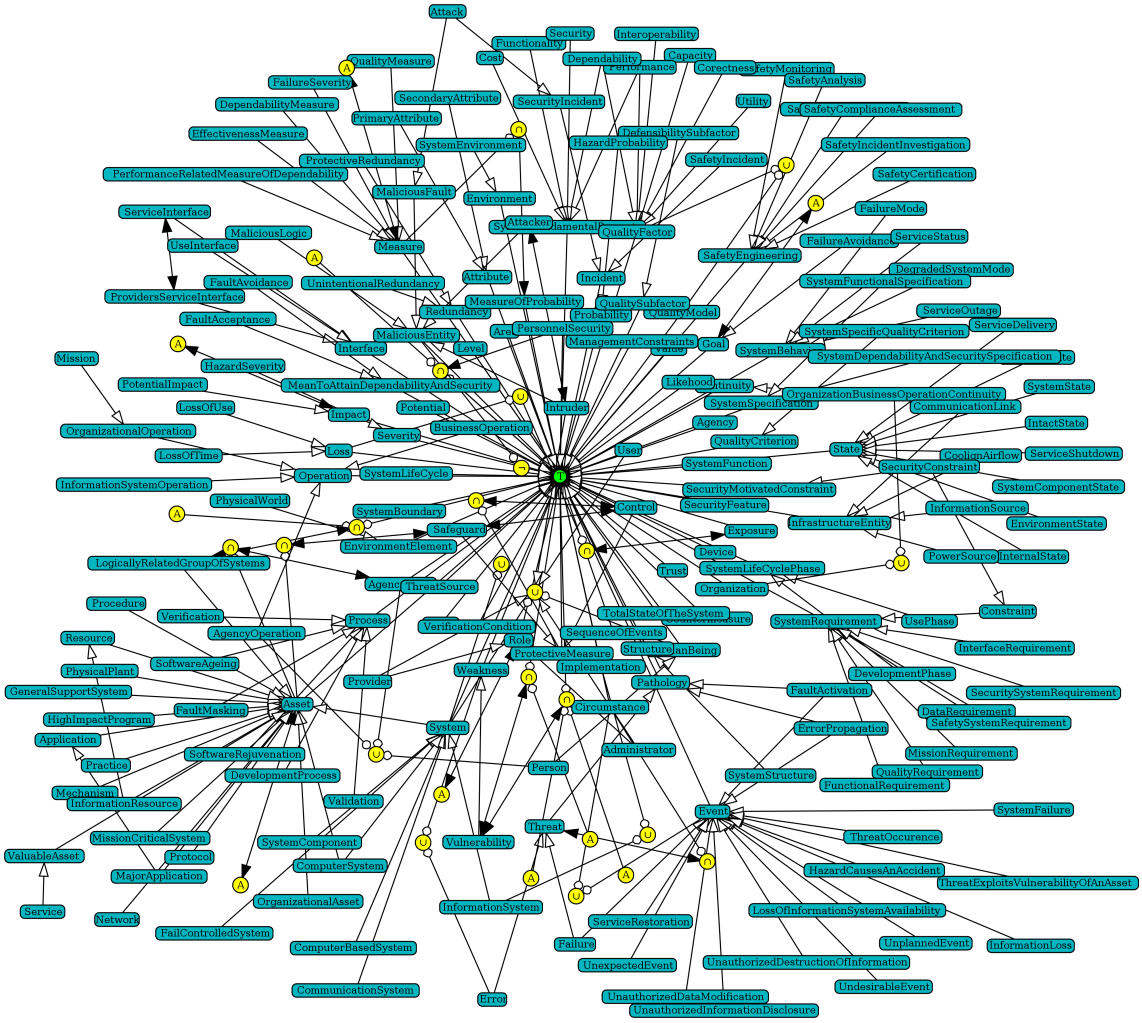
Konstrukcja Modułu bazowego

W pierwszej kolejności dokonano integracji ontologii opartych na słownikach ENISA oraz NIST. Dokładnie procedurę integracji tych ontologii opisano w rozdziale 4.6. Wynik tej integracji następnie połączono z ontologią opartą na książce Iana Sommerville'a. Łącznie w procesie integracji wykonano 1170 porównań na 153 klasach. W wyniku operacji scalenia powstała ontologia, którą następnie porównano z ręcznie utworzonym modułem podstawowym Ontologii bezpieczeństwa. Porównanie to wykazało kilka niespójności, które można zakwalifikować do jednej z trzech grup:

- błędy i niespójności w ontologiach składowych,
- zbyt szeroki zakres tematyczny ontologii składowych,
- brak części powiązań w ontologiach składowych,
- graniczne wartości przyjętych progów podobieństwa elementów ontologii.

Błędy i niespójności w ontologiach składowych W wyniku pracy algorytmu wykryto niespójności w ontologiach składowych Modułu bazowego Ontologii bezpieczeństwa. W ontologii opartej na książce Iana Sommerville'a pojęcie *Asset* („Aktywa"), opisane jest jako podklasa pojęcia *SystemResource* („ZasóbSystemowy"), które z kolei jest podklasą konceptu *Resource* („Zasób"). W pozostałych dwu ontologiach powiązanie to nie występuje. Analiza znaczeń tych trzech pojęć oraz struktury słownika WordNet wskazuje, że właściwa powinna być relacja stanowiąca, że pojęcie *Asset* jest najbardziej ogólne, pojęcie *Resource* zawiera się w nim znaczeniowo, a pojęcie *SystemResource* jest podklasą pojęcia *Resource*. Fakt ten znajduje swoje odzwierciedlenie w Module bazowym, gdzie występuje tylko ten drugi układ zależności pomiędzy tymi trzema pojęciami. Na podstawie tej analizy skorygowano zarówno finalną ontologię, jak i ontologię Sommerville'a, usuwając z nich zależność *owl:subClassOf(Asset,SystemResource)*. Obecność tej relacji w finalnej ontologii nie wpływało na proces jej konstrukcji, jednak tworzyła ona cykl, co w hierarchii wywnioskowanej obrazowało się jako tożsamość pojęć *Asset*, *Resource* oraz *SystemResource*.

Zbyt szeroki zakres tematyczny ontologii składowych Języki naturalne są językami wieloznacznymi, gdzie sens poszczególnych pojęć wynika nie tylko z samego pojęcia, ale również kontekstu w jakim został osadzony. Ze względu na trudność określenia kontekstu, w proponowanym algorytmie przyjęto dziedzinowość integrowanych ontologii. Zakres dziedziny uzależniony jest w dużej



Rysunek 6.18: Ontologia bezpieczeństwa – wizualizacja fragmentu ontologii

mierze od samej dziedziny. Wiele pojęć przyjmuje inne znaczenia nawet po niewielkim rozszerzeniu bazy znaczeniowej. Pojęcie „System” może oznaczać zarówno strukturę organizacyjną jak i np. system komputerowy. Mogłoby się wydawać, że znaczenia te pochodzą z rozłącznych dziedzin, jednak kiedy mówimy o systemach komputerowych w organizacjach znaczenie słowa „System” traci swoją jednoznaczność. Ze względu na poszerzony zakres tematyczny dziedziny konstruowanej ontologii, po jej integracji konieczne było więc wprowadzenie zmian uspójniających ją z zamierzeniami autora tworzącego Moduł bazowy poprzez ręczną integrację.

Algorytm błędnie powiązał pojęcie *System* z pojęciami *Agency*, *Organization* oraz *Structure*. Wszystkie te cztery pojęcia, w kontekście organizacyjnym, określają pewną formę organizacji, jednak w przypadku tworzonej ontologii, koncept *System* występuje w znaczeniu zasobu organizacji, a nie jej struktury. Podobnie błędnie powiązane ze sobą zostały pojęcia:

1. *Value* („Wartość”) oraz *Measure* („Miara”) – algorytm, kierując się strukturą słownika WordNet, błędnie zakwalifikował koncept *Value* jako „Wielkość”, przez co został on powiązany tożsamością z pojęciem *Measure*,

2. *Security* („Bezpieczeństwo”) oraz *Measure* („Miara”) – algorytm, kierując się strukturą słownika WordNet, błędnie zakwalifikował koncept *Measure* jako „Wartość bezpieczeństwa”, przez co został on powiązany tożsamością z pojęciem *Measure*,

3. *Event* („Zdarzenie") oraz *Impact* („Wpływ na") – algorytm, kierując się strukturą słownika WordNet, błędnie zakwalifikował koncept *Impact* jako „Uderzenie ciała fizycznego w inne ciało fizyczne", które słownik definiuje jako potomne względem pojęcia „Zdarzenie", przez co został on powiązany tożsamością z pojęciem *Measure*,

4. *Measure* („Miara") oraz *„Countermeasure"* („Przeciwdziałanie") – algorytm, kierując się strukturą słownika WordNet sklasyfikował pojęcie *„Countermeasure"* jako „Kontr-miara ", co spowodowało dodanie do wynikowej ontologii nieprawidłowej relacji *owl:subClassOf(Countermeasure, Measure)*.

Wszystkie nadmiarowe relacje zostały usunięte z docelowej ontologii Modułu bazowego przed przystąpieniem do drugiego etapu konstrukcji Ontologii bezpieczeństwa.

W ontologii opartej na książce Sommerville'a pojęcie *Event*, czyli *Zdarzenie*, określone jest jako rozłączne z pojęciem *Circumstance*, czyli *Okoliczność*. W słowniku WordNet z kolei, w większości ich znaczeń, pojęcia te są ze sobą silnie powiązane, przy czym pojęcie *Circumstance* oznaczone jest jako bardziej ogólne niż *Event*. W dziedzinie tworzonej ontologii bezpieczeństwa pojęcie *Circumstance* rozumiane jest jako pewne okoliczności składające się na pewne zdarzenie (*Event*). Okoliczności zdarzenia nie są więc logicznie szczególnymi wypadkami zdarzenia, a czymś co je definiuje. Relacja *owl:subClassOf(Event, Circumstance)* została więc usunięta z tworzonego Modułu bazowego.

Brak części powiązań w ontologiach składowych W wyniku wykonania zaproponowanego algorytmu otrzymano powiązania, które zostały nieświadomie pominięte w ręcznej integracji składowych Modułu bazowego. Automatycznie utworzona ontologia poszerzona została o powiązania tożsamości pomiędzy pojęciami *Severity* i *Level* (oba pojęcia zostały trafnie zakwalifikowane jako opisujące poziom zagrożenia) oraz *Vulnerability* i *Exposure* (oba pojęcia określają słabość zabezpieczeń systemów informatycznych). Oba te powiązania nie występują ani w ontologiach składowych Modułu bazowego ani w samym, utworzonym ręcznie, Module. Z kolei powiązanie podrzędności pomiędzy pojęciami *System* (pojęcie podrzędne) oraz *Asset* (pojęcie nadrzędne) występuje w ręcznie utworzonej ontologii, jednak nie zostało wygenerowane przez algorytm. Fakt ten należy tłumaczyć brakiem występowania powiązań pomiędzy tymi pojęciami zarówno w ontologiach składowych Modułu bazowego jak i w słowniku WordNet. Słownik WordNet ponadto nie definiuje pojęcia „system" w żaden sposób zbliżony do znaczenia „system komputerowy", w jakim został użyty w tworzonej ontologii.

Graniczne wartości przyjętych progów podobieństwa elementów ontologii Pojęcia *UnauthorizedAccess* oraz *UnauthorizedDestruction* zostały przez algorytm określone jako podobne z miarą podobieństwa wynoszącą $0,72$. Przekracza ona więc nieznacznie graniczną określoną doświadczalnie (tablica 4.2) i opisaną w rozdziale 4.4. Powiązanie to zostało usunięte z docelowej ontologii, a pojedynczy przypadek błędnej klasyfikacji nie powinien jednak świadczyć o konieczności zmiany wartości brzegowej pomiędzy tożsamością i rozłącznością elementów, zwłaszcza w obliczu poprawnej klasyfikacji jako podobne pojęć *Agency* oraz *Organization*, które zostały poprawnie zaklasyfikowane jako podobne na podstawie miary wynoszącej $0,715$.

Algorytm w trakcie swojej pracy wykonał 1170 porównań, z czego jedynie 13 okazało się błędnych (w tym część wynikająca z błędów konstrukcyjnych ontologii składowych), co stanowi 1,11% wszystkich wykonanych operacji.

Integracja modułów składowych Ontologii Bezpieczeństwa

W kolejnym kroku Moduł bazowy połączono z Modułem bezpieczeństwa i niezawodności. Wynik tej integracji wymagał korekty w przypadkach podobnych jak w trakcie konstrukcji Modułu bazowego. Z powstałej ontologii usunięto więc związki podrzędności pomiędzy pojęciami:

- *Event* oraz *Impact*,
- *Structure* oraz *System*,
- *Circumstance* oraz *Event*,
- *Security* oraz *Measure*,

oraz tożsamości pomiędzy pojęciami:

- *Organization* oraz *System*,
- *Operation* oraz *Procedure*,
- *Measure* oraz *Value*.

Ponadto algorytm błędnie sklasyfikował pojęcie *Event* jako podklasę pojęcia *Process*, *Redundancy* jako podklasę pojęć *Organization* i *Level* oraz tożsamość pojęcia *Operation* z pojęciem *Process*. W obu przypadkach błędy te wynikały z nakładania się znaczeń pojęć z różnych dziedzin.

Proces integracji Modułu bazowego z Modułem bezpieczeństwa i niezawodności wskazał też błąd w strukturze drugiej z łączonych ontologii. Moduł ten definiuje pojęcie „Podatność" (ang. *Vulnerability*) jako podklasę pojęcia „BłądWewnętrzny" (ang. *InternalFault*). Jednakże każdy rodzaj błędu jest podatnością systemu na atak. Powiązanie to zostało więc usunięte z Modułu bezpieczeństwa i niezawodności oraz zintegrowanej ontologii.

Tak powstała ontologia została zintegrowana z ostatnim modułem, czyli Modułem wymogów bezpieczeństwa, tworząc Ontologię bezpieczeństwa. Ponownie przeprowadzono korektę osiągniętego wyniku poprzez usunięcie kilku związków podrzędności pomiędzy pojęciami:

- *Event* oraz *Impact*,
- *Structure* oraz *System*,
- *Circumstance* oraz *Event*,
- *Process* oraz *Event*,

oraz tożsamości pomiędzy pojęciami *Organization* i *System* oraz *Measure* i *Value*.

Ponadto algorytm błędnie sklasyfikował pojęcia:

- *SecurityProcedure* jako tożsame z *SafetyEngineering* – podobieństwo pomiędzy pojęciami zostało określone jako $0,72$,
- *SecurityFeature* jako tożsame z *SafetyEngineering* – podobieństwo pomiędzy pojęciami zostało określone jako $0,70$,
- *SecurityFeature* jako tożsame z *QualityMeasure* – podobieństwo pomiędzy pojęciami zostało określone jako $0,89$,
- *EnvironmentElement* jako tożsame z *QualityFactor* – podobieństwo pomiędzy pojęciami zostało określone jako $0,72$.

Poza trzecim przypadkiem błąd ten wynika z granicznej wartości podobieństwa. Znaczącym błędem algorytm wykazał się w przypadku porównania *SecurityFeature* z *QualityMeasure*. W tym przypadku algorytm rozbił pojęcia na pojęcia składowe, czyli odpowiednio na: *security* i *feature* oraz *quality* i *measure*. Następnie algorytm określił maksymalne podobieństwo pomiędzy parami wyrazów: *security* oraz *quality* – $P = 0,375$, *security* oraz *measure* – $P = 0,932$, *feature* oraz

quality – $P = 0,849$, *feature* oraz *measure* – $P = 0,735$. Należy zauważyć, że wysokie wartości podobieństw są zbieżne z wcześniejszymi błędami algorytmu wynikającymi ze zbyt szerokiej dziedziny analizowanych ontologii. W wyniku tego, po zastosowaniu algorytmu znajdowania maksymalnego dopasowania w zbiorach dwudzielnych (Rozdział 4.3.3), otrzymano wysokie podobieństwo pomiędzy pojęciami. Jest to jednak jedyny taki przypadek spośród wszystkich wykonanych porównań. Analogicznie błędnie została wyznaczona tożsamość pomiędzy pojęciami „ŚrodowiskoSystemowe" (ang. *SystemEnvironment*) i „WymaganiaSystemowe" (ang. *SystemRequirement*) oraz „StrukturaSystemu" (ang. *SystemStructure*) i „WymaganiaSystemowe" (ang. *SystemRequirement*)

Podobny problem zachodzi w przypadku porównania pojęć „Miara" (ang. *Measure*) oraz „MiaraJakości" (ang. *QualityMeasure*). Drugie z pojęć w naturalny sposób powinno zostać zakwalifikowane jako uszczegółowienie pierwszego i przez to połączone z nim relacją podrzędności. Jednakże w strukturze słownika WordNet pojęcie „miara jakości" nie występuje, stąd ponownie zastosowano metodę węgierską jako sposób na określenie maksymalnego podobieństwa. Stąd w wynikowej ontologii powiązanie pomiędzy tymi dwoma pojęciami nie zostało uchwycone. Analogicznie algorytm nie odnalazł relacji podrzędności pomiędzy konceptami „ŚrodowiskoSystemowe" (ang. *SystemEnvironment*) i „Środowisko" (ang. *Environment*), „StrukturaSystemu" (ang. *SystemStructure*) i „Struktura" (ang. *Structure*), „Procedura" (ang. *Procedure*) i „ProceduraBezpieczeństwa" (ang. *SecurityProcedure*) oraz „Napastnik" (ang. *Attacker*) i „WrogaJednostka" (ang. *MaliciousEntity*).

Pomimo występowania w słowniku WordNet pojęcia „jednostka ludzka" (ang. *human being*), algorytm nie był w stanie powiązać pojęć „JednostkaLudzka" (ang. *HumanBeing*) i „Osoba" (ang. *Person*). Wynika to z faktu braku szczególnego zapisu pojęć wieloczłonowych w słowniku WordNet jak i brak jednoznacznej reprezentacji takich pojęć w języku naturalnym.

Ostatni proces integracji wskazał też błąd w konstrukcji Modułu wymogów bezpieczeństwa. Ontologia ta definiuje pojęcie „WymogiBezpieczeństwa" (ang. *SecurityRequirement*) jako składające się z czterech elementów: „Dostępności" (ang. *Availability*), „Poufności" (ang. *Confidentiality*), „Spójności" (ang. *Integrity*) oraz „BrakOdrzuceń" (ang. *Nonrepudiation*) rozumiane jako wymóg istnienia tych elementów, by możliwe było zapewnienie bezpieczeństwa. Ta, jak i pozostałe ontologie, wykorzystują jednak te same pojęcia w dosłownym ich rozumieniu, co wprowadza niejasność w ich interpretacji. W związku z tym wymienione pojęcia w kontekście wymogów bezpieczeństwa przemianowano dodając przyrostek „Wymóg" (ang. *Requirement*) (np. *AvailabilityRequirement*).

Sprawdzając, przy pomocy wnioskera HermiT, spójność pozyskanej ontologii bezpieczeństwa stwierdzono ponadto nieprawidłowości w ontologii Sommerville'a, wchodzącej w skład modułu bezpieczeństwa. Ontologia ta wprowadzała relację podrzędności pojęcia *Hazard* względem pojęcia *Event*. Błąd ten wynikał z błędnej konstrukcji pojęcia *Hazard*. Niewłaściwa relacja została usunięta w trakcie ręcznego tworzenia Ontologii bezpieczeństwa, jednak jej autorzy pominęli ten związek z ontologii Sommerville'a. Zastosowany algorytm umożliwił wychwycenie niespójności oraz korektę źródłowej ontologii.

Algorytm, konstruując Ontologię bezpieczeństwa, wykonał łącznie 1956 porównań, z czego 31 okazało się błędne (w tym wynikające z błędów konstrukcyjnych ontologii składowych). Podobnie jak w przypadku konstrukcji Modułu bazowego stanowi to 1,59% wszystkich wykonanych porównań.

Ontologia bezpieczeństwa

Obie ontologie bezpieczeństwa, zarówno utworzona poprzez integrację manualną jak i półautomatyczną, opartą na zaproponowanym algorytmie, są do siebie bardzo podobne. Ontologia integrowana półautomatycznie wymagała niewielkiej ilości korekt eliminujących dwuznaczności językowe.

Powyższy przykład obrazuje zależność algorytmu od poprawnej struktury integrowanych ontologii oraz od jakości wykorzystywanych w trakcie procesu łączenia zewnętrznych słowników. Niespójności w ontologiach składowych mają znaczący wpływ na ontologię wynikową. Błędy w ontologiach składowych prowadziły w niektórych przypadkach nawet do pozyskania niespójnej ontologii wynikowej. Niespójności te pozwalały jednak na szybkie znalezienie i korektę błędów w zewnętrznych

źródłach, obrazując przydatność zaproponowanego rozwiązania w procesie konstrukcji i integracji ontologii.

Ontologia integrowana za pomocą zaproponowanego algorytmu wymagała niewielkich korekt opisanych we wcześniejszych sekcjach tego rozdziału. Większość błędów algorytmu wynika ze zbyt szerokiej tematyki, co stoi w sprzeczności z założeniem dziedzinowości ontologii. Jednakże nawet w tym przypadku algorytm w 98,59% przypadków poprawnie określił związek pomiędzy pojęciami.

Należy zauważyć, że algorytm nie tylko wygenerował strukturę bardzo zbliżoną do utworzonej ręcznie przez człowieka, ale także pozwolił na wychwycenie błędów w ontologiach składowych Ontologii bezpieczeństwa. W trakcie prac nad algorytmem udało się wyeliminować niespójności i niejasności w opracowanych ontologiach składowych. Algorytm wygenerował również dodatkowe powiązania, które nie zostały uchwycone przez autorów oryginalnej Ontologii bezpieczeństwa, a które podnoszą jej spójność. Są to relacje tożsamości pomiędzy pojęciami:

- *Agency* oraz *Organization*,
- *Group* oraz *Organization*,
- *Area* oraz *Environment*,
- *Exposure* oraz *Vulnerability*,

oraz podrzędności pomiędzy pojęciami *Level* oraz *Severity*.

6.4 Ocena algorytmu

We wszystkich testowanych przypadkach proponowany algorytm spełnił pokładane w nim oczekiwania. Przeprowadzone badania wskazują jego poprawność zarówno w przypadku łączenia niewielkich ontologii, składających się z kilkudziesięciu pojęć, jak i dużych ontologii, złożonych z kilkuset konceptów i właściwości. We wszystkich testowanych przypadkach prawidłowość wykonanych porównań przekraczała 98%. W każdym przypadku poprawność ontologii wejściowych gwarantowała wytworzenie spójnej ontologii wyjściowej. Algorytm upraszcza również eliminację błędów czy braków zarówno w opisie, jak i strukturze elementów ontologii wejściowych, poprzez uproszczenie analizy związków pomiędzy pojęciami integrowanych źródeł. Wraz z rozwojem słowników semantycznych, takich jak WordNet, czy użytych algorytmów pomocniczych, uzyskiwana jakość algorytmu może jeszcze wzrosnąć.

Pesymistyczna złożoność obliczeniowa proponowanego rozwiązania, gdy każdy element ontologii A będzie podobny do każdego elementu ontologii B, wynosi $O(ab)$, gdzie a to liczba klas i bytów ontologii A, b to liczba klas i bytów ontologii B.

Zaletą algorytmu jest również jego prostota i przenośność. W przeciwieństwie do innych systemów takich jak PROMPT czy FalconAO, jego implementacja w postaci biblioteki w języku Java umożliwia zastosowanie algorytmu w dowolnym systemie operującym na ontologiach w języku OWL bez konieczności zapewniania interakcji z człowiekiem. Algorytm może być więc zastosowany również np. w systemach agentowych, czy usługach sieciowych, do zautomatyzowania procesu integracji ontologii, a tym samym zapewnienia większego poziomu współpracy pomiędzy różnymi usługami czy systemami.

Rozdział 7

Wnioski końcowe

W niniejszej rozprawie przedstawiono zaproponowany przez autora algorytm łączenia i odwzorowywania ontologii dziedzinowych, który może znaleźć zastosowanie w procesie integracji wiedzy zapisanej w postaci ontologii. Do najważniejszych wyników rozprawy należą:

- zdefiniowanie pojęcia i stopni integracji ontologii,
- zaproponowanie nowego algorytmu integracji wiedzy opartego na analizie leksykalnej i semantycznej ontologii,
- zaproponowanie metody wykorzystania wiedzy zawartej w leksykonach w procesie integracji wiedzy,
- opracowanie systemu tworzenia i integracji ontologii w języku OWL.

Zaproponowany algorytm wykorzystuje wiele skutecznych procedur łączenia oraz odwzorowywania ontologii dziedzinowych, a także umożliwia kompleksową analizę ontologii wejściowych, jak również ontologii wyjściowej złożonej z tych pierwszych.

W rozdziale trzecim dokonano przeglądu istniejących technik łączenia i odwzorowywania ontologii, a w pierwszej części rozdziału czwartego przedstawiono warunki niezbędne do realizacji tego procesu. W rozdziałach tych zdefiniowano podstawowe pojęcia związane z procedurami łączenia i odwzorowywania ontologii, a także zaproponowano miary podobieństwa pomiędzy konceptami wchodzącymi w skład integrowanych ontologii. Dokonano też przeglądu miar i algorytmów pomocniczych oraz wybrano te, na których oparto proponowane rozwiązanie zapewniające spójność i poprawność analizowanych ontologii, porównywalną z możliwościami analizy realizowanej przez człowieka, oraz wysoką efektywność działania, znacznie wyższą od analizy ręcznej.

W drugiej części rozdziału czwartego zaprezentowano algorytm automatycznego łączenia i odwzorowywania ontologii, a rozdział piąty prezentuje rozszerzenie systemu OCS o implementację tego algorytmu. Rozdział szósty prezentuje metodologię wytwarzania ontologii dziedzinowych wspierającą możliwości ich przyszłej integracji.

W ten sposób wykazano tezy rozprawy doktorskiej dotyczące zarówno możliwości budowy takiego algorytmu, jak również jego praktycznej realizowalności w postaci systemu OCS. Co więcej wykorzystanie takowego narzędzia wspomaga oryginalną metodologię wytwarzania ontologii dziedzinowych upraszczającą ich przyszłą integrację.

Opracowany algorytm zaimplementowany został w systemie OCS (*Ontology Creation System*), umożliwiającym zespołowe tworzenie ontologii oraz ich składowanie i wersjonowanie. System ten operuje na ontologiach w języku OWL. Rozszerzenie systemu OCS o implementację proponowanego algorytmu umożliwia szybsze i bardziej dokładne tworzenie nowych ontologii poprzez integrację i rozbudowę już istniejących rozwiązań. System OCS wraz z implementacją algorytmu dostępny

jest na otwartej licencji, co umożliwi dalszy rozwój samego systemu, jak i proponowanego algorytmu.

Niniejsza rozprawa nie kończy definitywnie prac nad rozwojem procedur łączenia i odwzorowywania ontologii. Dalsze możliwości udoskonalenia tego typu algorytmów dotyczące poprawy technik analizy języka naturalnego i określania znaczenia słów, jak również leksykonów takich jak np. słownik WordNet, pozwolą na lepsze określanie kontekstu, w jakim osadzone są pojęcia wchodzące w skład integrowanych ontologii, a tym samym poprawi jakość konstruowanych ontologii wynikowych.

Spis rysunków

Spis tablic

Spis algorytmów

Bibliografia

[1] M. Abu Helou, A. Abid. Semantic measures based on WordNet using multiple information sources. *Proceedings of International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management IC3K2010*. INSTICC, 2010.

[2] R. Anderson. Inżynieria zabezpieczeń, 2005.

[3] G. Antoniou, F. Harmelen. Web ontology language: Owl. *Handbook on ontologies*, strony 91–110, 2009.

[4] Apache Subversion. Subversion. 2008. [Online; przeglądano 16-06-2011].

[5] R. Araújo, H.S. Pinto. Semilarity: Towards a model-driven approach to similarity. *International Workshop on Description Logics*, strony 155–162. Citeseer, 2007.

[6] R. Araújo, H.S. Pinto. Towards semantics-based ontology similarity. *Proceedings of the Workshop on Ontology Matching (OM2007) at ISWC/ASWC2007, Busan, South Korea*. Citeseer, 2007.

[7] A. Avizienis, J.C. Laprie, B. Randell, C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1):11–33, 2004.

[8] F. Baader. *The description logic handbook: theory, implementation, and applications*. Cambridge Univ Pr, 2003.

[9] S. Banerjee. *Adapting the Lesk algorithm for word sense disambiguation to WordNet*. Praca doktorska, Citeseer, 2002.

[10] S. Banerjee, T. Pedersen. Extended gloss overlaps as a measure of semantic relatedness. *International Joint Conference on Artificial Intelligence*, wolumen 18, strony 805–810. Citeseer, 2003.

[11] S. Banerjee, T. Pedersen. An adapted Lesk algorithm for word sense disambiguation using WordNet. *Computational Linguistics and Intelligent Text Processing*, strony 117–171, 2010.

[12] R.J. Bayardo, J.D. Pehoushek. Counting models using connected components. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, strony 157–162. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2000.

[13] S. Bechhofer, R. Volz, P. Lord. Cooking the semantic web with the owl api. *The Semantic Web-ISWC 2003*, strony 659–675, 2003.

[14] D. Beckett, B. McBride. Rdf/xml syntax specification (revised). *W3C recommendation*, 10, 2004.

[15] T. Bemers-Lee, J. Hendler, O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.

[16] M.K. Bergman. Basing UMBEL's Backbone on OpenCyc, Part 4 of 4 on Foundations to UMBEL. `http://www.mkbergman.com/433/basing-umbels-backbone-on-opencyc/`, 2008. [Online; przeglądano 12-10-2010].

[17] M.K. Bergman, F. Giasson. Upper Mapping and Binding Exchange Layer, A lightweight, subject concept reference structure for the Web. `http://www.umbel.org/`, 2007. [Online; przeglądano 12-10-2010].

[18] T. Boiński. Architektura portalu dziedzinowego. *Praca zbiorowa Katedry Architektury Systemów Komputerowych KASKBOOK*, strony 81–92, 2008.

[19] T. Boiński. Budowa ontologii usług dla potrzeb wyszukiwania. *Praca zbiorowa Katedry Architektury Systemów Komputerowych KASKBOOK*, strony 47–58, 2010.

[20] T. Boiński, Ł. Budnik, A. Jakowski, J. Mroziński, K. Mazurkiewicz. OCS – domain oriented ontology creation system. *Polish Journal of Environmental Studies*, 18(3B):35–38, 2009.

[21] T. Boiński, A. Jaworska, R. Kleczkowski, P. Kunowski. Ontology visualization. *Information Technology (ICIT), 2010 2nd International Conference on*, strony 17–20. IEEE, 2010.

[22] T. Boiński, A. Jaworska, R. Kleczkowski, P. Kunowski, J. Szamański. Zespołowa budowa ontologii z wykorzystaniem systemu OCS oraz edytora Protégé. *Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej*, 19:101–105, 2010.

[23] T. Boiński, P. Orłowski, P. Szpryngier. Inżynieria ontologii dla potrzeb integracji systemów. *Praca zbiorowa Katedry Architektury Systemów Komputerowych KASKBOOK (w publikacji)*, 2012.

[24] T. Boiński, P. Orłowski, P. Szpryngier, H. Krawczyk. Influence and selection of basic concepts on ontology design. *Proceedings of KEOD2010 of the 2nd International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, strony 364–369. INSTICC, 2010.

[25] T. Boiński, P. Orłowski, J. Szymański, H. Krawczyk. Security ontology construction and integration. *Proceedings of KEOD2011 of the 3nd International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, strony 369–374. INSTICC, 2011.

[26] D. Bokal, B. Bresar, J. Jerebic. A generalization of hungarian method and hall's theorem with applications in wireless sensor networks. *Arxiv preprint arXiv:0911.1269*, 2009.

[27] A. Borgida, T. Walsh, H. Hirsh. Towards measuring similarity in description logics. *Working Notes of the International Description Logics Workshop*, wolumen 147. Citeseer.

[28] P. Bouquet, J. Euzenat, E. Franconi, L. Serafini, G. Stamou, S. Tessaris. D2.2.1 Specification of a common framework for characterizing alignment. *Knowledge Web project EU-IST-2004-507482, realizing the semantic web*, 2004.

[29] D. Brickley. Rdf vocabulary description language 1.0: Rdf schema. *http://www.w3.org/tr/rdf-schema/*, 2004.

[30] D. Brickley, R.V. Guha. Resource description framework (RDF) schema specification 1.0. *W3C Candidate Recommendation*, 27:2001–03, 2000.

[31] D. Brickley, L. Miller, V.S. FOAF. 0.98–namespace document 9 august 2010–marco polo edition. `http://xmlns.com/foaf/spec/`. [Online; przeglądano 31-01-2012].

[32] H. Bulskov, R. Knappe, T. Andreasen. On measuring similarity for conceptual querying. *Flexible Query Answering Systems*, strony 100–111, 2002.

[33] R.E. Burkard, M. Dell'Amico, S. Martello. *Assignment problems*. Society for Industrial Mathematics, 2009.

[34] Z. Cackowski, J. Kmita, K. Szaniawski, PJ Smoczyński. Filozofia a nauka. *Zarys encyklopedyczny, Zakład Narodowy im. Ossolińskich, Wydawnictwo Polskiej Akademii Nauk, Wrocław–Warszawa–Kraków–Gdańsk–Łódź*, 1987.

[35] B. Collins-Sussman, B.W. Fitzpatrick, C.M. Pilato. *Version control with subversion*. O'Reilly Media, Inc., 2004.

[36] T.M. Cover, J.A. Thomas, J. Wiley, i in. *Elements of information theory*, wolumen 6. Wiley Online Library, 1991.

[37] M. Croitoru, B. Hu, S. Dashmapatra, P. Lewis, D. Dupplaw, L. Xiao. A Conceptual Graph Based Approach to Ontology Similarity Measure. *Conceptual Structures: Knowledge Architectures for Smart Applications*, strony 154–164, 2007.

[38] R. Culmone, G. Rossi, E. Merelli. An ontology similarity algorithm for BioAgent. *NETTAB Workshop on Agents nd Bioinformtics, Bologna*. Citeseer, 2002.

[39] Cycorp, Inc. about Cycorp. `http://cyc.com/cyc/company/about`, 2001. [Online; przeglądano 06-10-2010].

[40] Cycorp, Inc. FACTory game. `http://game.cyc.com/helpfiles/HowToPlay.html`, 2005. [Online; przeglądano 12-10-2010].

[41] Cycorp, Inc. Cycorp, Inc. `http://www.cyc.com/cyc`, 2010. [Online; przeglądano 06-10-2010].

[42] M. d'Aquin, A. Gangemi, E. Motta, M. Dzbor, P. Haase, M. Erdmann. Neon tool support for building ontologies by reuse. Citeseer, 2009.

[43] J. Davies, F. van Harmelen, D. Fensel. *Towards the semantic web: ontology-driven knowledge management*. John Wiley & Sons, Inc. New York, NY, USA, 2002.

[44] A. De Nicola, M. Missikoff, R. Navigli. A software engineering approach to ontology building. *Information Systems*, 34(2):258–275, 2009.

[45] M. Dean, D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein. OWL web ontology language 1.0 reference. *W3C Working Draft*, 29, 2002.

[46] R. Dieng, S. Hug. Comparison of personal ontologies represented through conceptual graphs. *Proceedings of ECAI*, wolumen 98, strony 341–345. Citeseer, 1998.

[47] A.H. Doan, J. Madhavan, P. Domingos, A. Halevy. Learning to map between ontologies on the semantic web. *Proceedings of the 11th international conference on World Wide Web*, strony 662–673. ACM, 2002.

[48] M. Ehrig, Y. Sure. Ontology mapping-an integrated approach. *The Semantic Web: Research and Applications*, strony 76–91, 2004.

[49] ENISA. Risk management: implementation principles and inventories for risk management/risk assessment methods and tools. Technical report, 2006.

[50] Enisa. Enisa: a European Union Agency - Glossary of Risk Management. 2010. [Online; przeglądano 01-12-2010].

[51] J. Euzenat. Eon ontology alignment contest. 2004. [Online; przeglądano 27-01-2012].

[52] J. Euzenat. An api for ontology alignment (version 2.1). *INRIA Rhone-Alpes*, 2006.

[53] J. Euzenat, P. Valtchev. Similarity-based ontology alignment in OWL-lite. *ECAI 2004: 16th European Conference on Artificial Intelligence, August 22-27, 2004, Valencia, Spain: including Prestigious Applicants [sic] of Intelligent Systems (PAIS 2004): proceedings*, strona 333. Ios Pr Inc, 2004.

[54] C. Fellbaum. *WordNet: An electronic lexical database*. The MIT press, 1998.

[55] D. Fensel, C. Bussler, Y. Ding, V. Kartseva, M. Klein, M. Korotkiy, B. Omelayenko, R. Siebes. Semantic web application areas. *NLDB Workshop*. Citeseer, 2002.

[56] S. Fenz, T. Pruckner, A. Manutscheri. Ontological mapping of information security best-practice guidelines. *Business Information Systems*, strony 49–60. Springer, 2009.

[57] M. Fernandez, A. Gomez-Perez, N. Juristo. Methontology: from ontological art towards ontological engineering. *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, strony 33–40, 1997.

[58] D.G. Firesmith. A Taxonomy of safety-related requirements. *International Workshop on High Assurance Systems (RHAS'05)*, 2005.

[59] D.G. Firesmith. A taxonomy of security-related requirements. *International Workshop on High Assurance Systems (RHAS'05)*. Citeseer, 2005.

[60] A. Frank. On kuhn's hungarian method – a tribute from hungary. *Naval Research Logistics (NRL)*, 52(1):2–5, 2005.

[61] B. Ganter, R. Wille. Formal Concept Analysis: Mathematical Foundations, 1999.

[62] J.H. Gennari, M.A. Musen, R.W. Fergerson, W.E. Grosso, M. Crubézy, H. Eriksson, N.F. Noy, S.W. Tu. The evolution of Protégé: an environment for knowledge-based systems development. *International Journal of Human-Computer Studies*, 58(1):89–123, 2003.

[63] M. Gilleland. Levenshtein Distance, in Three Flavors. `http://www.merriampark.com/ld.htm`, 2011. [Online; przeglądano 31-05-2011].

[64] R. Goclenius. *Lexicon philosophicum quo tamquam clavae philosophiae fors aperiuntur*. 1613.

[65] K. Goczyła. *Ontologie w systemach informatycznych*. Akademicka Oficyna Wydawnicza EXIT, 2011.

[66] K. Goczyła, T. Grabowska. Metoda ELPAR łączenia ontologii oparta na ich kartograficznej reprezentacji. *I Krajowa Konferencja Naukowa-Technologie Przetwarzania Danych*, 2005.

[67] K. Goczyla, T. Grabowska, W. Waloszek, M. Zawadzki. The cartographer algorithm for processing and querying description logics ontologies. *Advances in Web Intelligence*, strony 163–169, 2005.

[68] K. Goczyła, W. Waloszek. Topologiczna analiza ontologii opartych na logice opisowej. *Bazy Danych: Aplikacje i Systemy*, strony 191–197, 2005.

[69] A. Gómez-Pérez, O. Corcho, M. Fernandez-Lopez. *Ontological Engineering*. Springer-Verlag, London, Berlin, 2002.

[70] T.R. Gruber, i in. A translation approach to portable ontology specifications. *Knowledge acquisition*, 5:199–199, 1993.

[71] B. Guttman, E.A. Roback. *An introduction to computer security: the NIST handbook*. DIANE Publishing, 1995.

[72] A. Herzog, N. Shahmehri, C. Duma. An ontology of information security. 2009.

[73] M. Horridge, S. Bechhofer, O. Noppens. Igniting the owl 1.1 touch paper: The owl api. *Proc. OWL-ED*, 258, 2007.

[74] I. Horrocks, i in. Daml+oil: A description logic for the semantic web. *IEEE Data Engineering Bulletin*, 25(1):4–9, 2002.

[75] E. Hovy. Combining and standardizing large-scale, practical ontologies for machine translation and other uses. *Proceedings of the 1st International Conference on Language Resources and Evaluation (LREC)*, strony 535–542. Citeseer, 1998.

[76] IEEE P1600.1. Standard Upper Ontology Working Group (SUO WG). `http://suo.ieee.org/index.html`, grudzień 2003. [Online; przeglądano 06-10-2010].

[77] IEEE P1600.1. Standard Upper Ontology Working Group (SUO WG), Suggested Upper Merged Ontology. `http://suo.ieee.org/SUO/SUMO/index.html`, grudzień 2003. [Online; przeglądano 06-10-2010].

[78] International Atomic Energy Agency. IAEA Safety Glossary Terminology Used In Nuclear Safety And Radiation Protection. `www-pub.iaea.org/MTCD/publications/PDF/Pub1290_web.pdf`, 2007. [Online; przeglądano 21-12-2011].

[79] A. Islam, D. Inkpen. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10, 2008.

[80] ISO/IEC. Information technology – Security techniques – Management of information and communications technology security – Part 1: Concepts and models for information and communications technology security management. `http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39066`, 2004. [Online; przeglądano 21-12-2011].

[81] N. Jian, W. Hu, G. Cheng, Y. Qu. Falcon-AO: Aligning ontologies with falcon. *Integrating Ontologies Workshop Proceedings*, strona 85. Citeseer, 2005.

[82] J.J. Jiang, D.W. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. *Arxiv preprint cmp-lg/9709008*, 1997.

[83] E. Jiménez-Ruiz, B.C. Grau, I. Horrocks, R. Berlanga. Logic-based ontology integration using contentmap. *Proc. of XIV Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2009)*, strony 316–319, 2009.

[84] B. Kaczorowski. *Nowa encyklopedia powszechna PWN*. Wydawnictwo Naukowe PWN, 2004.

[85] Y. Kalfoglou, M. Schorlemmer. Ontology mapping: the state of the art. *The knowledge engineering review*, 18(01):1–31, 2003.

[86] P. Kapłański. *Logika opisowa jako język modelowania oprogramowania.*, strony 157–166. PWNT, 2008.

[87] A. Kim, J. Luo, M. Kang. Security ontology for annotating resources. *On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE*, strony 1483–1499, 2005.

[88] R. Kissel. Glossary of key information security terms. *Glossary, National Institute of Standards and Technology, US Department of Commerce*, 2006.

[89] G. Klyne, J.J. Carroll. Resource description framework (rdf): Concepts and abstract syntax. *Changes*, 10(February):1–20, 2004.

[90] H. Knublauch. Protégé-owl api programmer's guide. 2009. [Online; przeglądano 17-06-2011].

[91] D. Koller, A. Levy, A. Pfeffer. P-CLASSIC: A tractable probablistic description logic. *Proceedings of the National Conference on Artificial Intelligence*, strony 390–397. Citeseer, 1997.

[92] H.W. Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[93] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. *Proceedings of the 5th annual international conference on Systems documentation*, strony 24–26. ACM, 1986.

[94] V.I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, wolumen 10, strony 707–710, 1966.

[95] Y. Li, Z.A. Bandar, D. McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on knowledge and data engineering*, strony 871–882, 2003.

[96] D. Lin. Principle-based parsing without overgeneration. *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, strony 112–120. Association for Computational Linguistics, 1993.

[97] D. Lin. An information-theoretic definition of similarity. *Proceedings of the 15th International Conference on Machine Learning*, wolumen 1, strony 296–304. Citeseer, 1998.

[98] P.W. Lord, R.D. Stevens, A. Brass, C.A. Goble. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics*, 19(10):1275, 2003.

[99] A. Maedche, S. Staab. *Comparing ontologies-similarity measures and a comparison study.* Institute AIFB, University of Karlsruhe, 2001.

[100] A. Maedche, S. Staab. Measuring similarity between ontologies. *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, strony 15–21, 2002.

[101] D.L. McGuinness, R. Fikes, J. Rice, S. Wilder. An environment for merging and testing large ontologies. *International Conference on Principles of Knowledge Representation and Reasoning*, strony 483–493. Citeseer, 2000.

[102] D.L. McGuinness, F. Van Harmelen, i in. OWL web ontology language overview. *W3C recommendation*, 10:2004–03, 2004.

[103] E. Miller, R. Swick. An Overview of W3C Semantic Web Activity. *Bulletin of the American Society for Information Science and Technology*, 29(4):8–11, 2003.

[104] G.A. Miller, W.G. Charles. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28, 1991.

[105] J. Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.

[106] D. Nardi, R.J. Brachman. An introduction to description logics. *The description logic handbook: theory, implementation, and applications*, strony 1–40, 2003.

[107] NeOn Project. NeOn Book. 2010. [Online; przeglądano 09-10-2010].

[108] I. Niles. Origins of the IEEE standard upper ontology. *Working Notes of the IJCAI-2001 Workshop on the IEEE Standard Upper Ontology*, strony 37–42. Citeseer, 2001.

[109] I. Niles, A. Pease. Towards a standard upper ontology. *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, strony 2–9. ACM, 2001.

[110] N. Noy, R. Fergerson, M. Musen. The knowledge model of Protege-2000: Combining interoperability and flexibility. *Knowledge Engineering and Knowledge Management Methods, Models, and Tools*, strony 69–82, 2000.

[111] N.F. Noy. Semantic integration: a survey of ontology-based approaches. *ACM Sigmod Record*, 33(4):65–70, 2004.

[112] N.F. Noy, D.L. McGuinness, i in. Ontology development 101: A guide to creating your first ontology, 2001.

[113] N.F. Noy, M.A. Musen. Algorithm and tool for automated ontology merging and alignment. *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00). Available as SMI technical report SMI-2000-0831*, 2000.

[114] N.F. Noy, M.A. Musen. The prompt suite: interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.

[115] ontologydesignpatterns.org. Ontology Design Patterns. `http://ontologydesignpatterns.org/wiki/Main_Page`, listopad 2011. [Online; przeglądano 21-12-2011].

[116] Oracle. Java Web Start Technology. 2005. [Online; przeglądano 17-06-2011].

[117] Z. Pawlak. Rough Sets. *International Journal of Information and Computer Science*, strony 341–356, 1982.

[118] A. Pease. Suggested Upper Merged Ontology (SUMO). `http://www.ontologyportal.org/index.html`, maj 2010. [Online; przeglądano 06-10-2010].

[119] Piotrowski, M. Zarządzanie ryzykiem cz. 1. `http://www.e-ochronadanych.pl/a,297,zarzadzanie-ryzykiem-cz-i-.html`, czerwiec 2008. [Online; przeglądano 21-12-2011].

[120] Prefuse. Prefuse. 2009. [Online; przeglądano 16-06-2011].

[121] Princeton University. WordNet Database Statistics. `http://wordnet.princeton.edu/wordnet/man/wnstats.7WN.html`, 2010. [Online; przeglądano 22-11-2010].

[122] P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of artificial intelligence research*, 11(95):130, 1999.

[123] R. Richardson, A.F. Smeaton, J. Murphy. Using WordNet as a knowledge base for measuring semantic similarity between words. *Proceedings of AICS Conference. Trinity College, Dublin.* Citeseer, 1994.

[124] M.A. Rodríguez, M.J. Egenhofer. Determining semantic similarity among entity classes from different ontologies. *IEEE transactions on knowledge and data engineering*, strony 442–456, 2003.

[125] D. Schober, W. Kusnierczyk, S.E. Lewis, J. Lomax, i in. Towards naming conventions for use in controlled vocabulary and ontology engineering. *Proceedings of BioOntologies SIG, ISMB07*, strony 29–32, 2007.

[126] M. Schuemacher. *Security engineering with patterns: origins, theoretical model, and new applications*. Springer-Verlag, 2003.

[127] E. Sirin, B. Parsia, B.C. Grau, A. Kalyanpur, Y. Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: science, services and agents on the World Wide Web*, 5(2):51–53, 2007.

[128] I. Sommerville. Software Engineering. 8th. *Harlow, UK: Addison-Wesley*, 2006.

[129] J.F. Sowa. Conceptual Graphs. 2005. [Online; przeglądano 21-03-2011].

[130] J.F. Sowa. Conceptual graphs. *Foundations of Artificial Intelligence*, 3:213–237, 2008.

[131] Stanford Center for Biomedical Informatics Research. Collaborative Protégé. 2009. [Online; przeglądano 16-06-2011].

[132] Stanford University School of Medicine. Stanford Center for Biomedical Informatics Research. 2010. [Online; przeglądano 09-10-2010].

[133] W. Stróżewski. *Ontologia*. ZNAK, 2006.

[134] M. Suárez-Figueroa, A. Gómez-Pérez, B. Villazón-Terrazas. How to write and use the Ontology Requirements Specification Document. *On the Move to Meaningful Internet Systems: OTM 2009*, strony 966–982, 2009.

[135] M.C. Suárez-Figueroa. D5. 3.1 NeOn Development Process and Ontology Life Cycle.

[136] MC Suárez-Figueroa, i in. D5. 4.2: Revision and extension of the neon methodology for building contextualized ontology networks. *NeOn project. http://www. neon-project. org*, 2009.

[137] Y. Sure, S. Staab, R. Studer. *Handbook on Ontologies*. 2009.

[138] M. Thurley. sharpSAT–Counting Models with Advanced Component Caching and Implicit BCP. *Theory and Applications of Satisfiability Testing-SAT 2006*, strony 424–429, 2006.

[139] A. Tversky. Features of Similarity. *Preference, Belief, and Similarity*, strony 7–46.

[140] G. Varelas, E. Voutsakis, P. Raftopoulou, E.G.M. Petrakis, E.E. Milios. Semantic similarity methods in wordNet and their application to information retrieval on the web. *Proceedings of the 7th annual ACM international workshop on Web information and data management*, strony 10–16. ACM, 2005.

[141] W3C. W3C Semantic Web Activity. `http://www.w3.org/2001/sw/`, 2001. [Online; przeglądano 18-10-2010].

[142] W3C, Heflin J. OWL Web Ontology Language Use Cases and Requirements. `http://www.w3.org/TR/webont-req/`, luty 2004. [Online; przeglądano 21-12-2011].

[143] Wikimedia Foundation, Inc. Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/Main_Page`, 2001. [Online; przeglądano 06-10-2010].

[144] Z. Wu, M. Palmer. Verbs semantics and lexical selection. *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, strony 133–138. Association for Computational Linguistics, 1994.

[145] Y. Zhao, W. Halang. Rough concept lattice based ontology similarity measure. *Proceedings of the 1st international conference on Scalable information systems*, strona 15. ACM, 2006.