

**Akademia Górniczo-Hutnicza  
im. Stanisława Staszica w Krakowie**

---

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki  
Katedra Automatyki



**PRACA MAGISTERSKA**

**SZYMON ŚWIERKOSZ**

**INTERNETOWY ROBOT WYSZUKUJĄCY STRONY  
DOTYCZĄCE WYBRANEJ DZIEDZINY WIEDZY NA  
PODSTAWIE ZAPYTAŃ W JEZYKU NATURALNYM.**

PROMOTOR:  
dr Adrian Horzyk

Kraków 2010

## **OŚWIADCZENIE AUTORA PRACY**

OŚWIADCZAM, ŚWIADOMY ODPOWIEDZIALNOŚCI KARNEJ ZA POŚWIADCZENIE NIEPRAWDY, ŻE NINIEJSZĄ PRACĘ DYPLOMOWĄ WYKONAŁEM OSOBIŚCIE I SAMODZIELNIE, I NIE KORZYSTAŁEM ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W PRACY.

.....

PODPIS

**AGH**  
**University of Science and Technology in Krakow**

---

Faculty of Electrical Engineering, Automatics, Computer Science and Electronics  
Institute of Automatics



**AGH**

**MASTER OF SCIENCE THESIS**

**SZYMON ŚWIERKOSZ**

**INTERNET ROBOT SEARCHING FOR WEB PAGES REFERING  
TO THE CHOSEN SCOPE OF KNOWLEDGE USING QUERIES IN  
NATURAL LANGUAGE.**

SUPERVISOR:  
Adrian Horzyk Ph.D

Krakow 2010

Serdecznie dziękuję rodzinie i promotorowi za wsparcie podczas pisania tej pracy oraz cenne wskazówki.

## Spis treści

<b>1. Wstęp</b> .....	7
1.1. Cele pracy.....	7
1.2. Zawartość pracy .....	7
<b>2. Wprowadzenie</b> .....	8
2.1. Przetwarzanie języka naturalnego.....	8
2.2. Wyszukiwarki internetowe.....	9
2.3. Chatbot.....	11
2.3.1. Zastosowania .....	11
2.3.2. Sposób działania.....	11
2.4. Istniejące wyszukiwarki.....	12
2.4.1. KtoCo.pl .....	13
2.4.2. Hipisek.pl .....	14
2.5. Wikipedia, wolna encyklopedia .....	15
2.5.1. Historia i rozwój.....	15
2.5.2. Najważniejsze zasady.....	15
2.5.3. Społeczność edytorów oraz wiarygodność projektu .....	16
2.5.4. Projekty siostrzane.....	17
2.5.5. Oprogramowanie .....	17
<b>3. Opis stworzonego rozwiązania</b> .....	19
3.1. Założenia projektowe .....	19
3.2. Architektura systemu .....	20
3.3. Implementacja .....	21
3.4. Słownik .....	23
3.4.1. Formy fleksyjne .....	23
3.4.2. Synonimy.....	24
3.5. Chatbot.....	25
3.5.1. Modelowanie dialogu .....	25
3.5.2. Przetwarzanie wstępne wypowiedzi.....	28
3.5.3. Dopasowywanie wzorców.....	29
3.5.4. Odpowiadanie.....	30
3.6. Wyszukiwarka.....	31

---

3.6.1. Transformacja zapytań .....	33
3.6.2. Klasyfikacja dokumentów .....	35
3.7. Praca aplikacji .....	37
<b>4. Prezentacja oraz ocena działania aplikacji .....</b>	<b>39</b>
4.1. Interfejs użytkownika .....	39
4.2. Wybrane rozmowy .....	40
4.3. Porównanie z innymi serwisami .....	44
<b>5. Podsumowanie i wnioski .....</b>	<b>46</b>
<b>A. Wykorzystane biblioteki oraz oprogramowanie .....</b>	<b>47</b>

# 1. Wstęp

Powstanie sieci Internet znacznie ułatwiło swobodną wymianę informacji pomiędzy ludźmi znajdującymi się na całym świecie. Każdy użytkownik może publikować własne materiały zakładając bezpłatnie własną stronę internetową. Rosnąca popularność tego medium spowodowała gwałtowny wzrost ilości publikowanej treści. Zaistniała potrzeba stworzenia mechanizmu, który pozwalałby użytkownikowi na dotarcie do stron, które znajdują się w obszarze jego zainteresowań. Zadanie to dodatkowo utrudnia fakt, że nie istnieje spis wszystkich stron internetowych w sieci.

Aby sprostać temu wyzwaniu zaczęto konstruować wyspecjalizowane programy – roboty internetowe (ang. *web crawlers*), których zadaniem było przeglądanie sieci. Na podstawie łączy między stronami kolejne dokumenty są pobierane, analizowane oraz indeksowane w bazie. Istniejące obecnie rozwiązania ogólnego przeznaczenia takie jak Google, Yahoo! oraz Bing pozwalają użytkownikowi na określenie poszukiwanej informacji w specjalnym języku zapytań. Niestety nie jest on językiem naturalnym – słowa kluczowe należy dobierać w odpowiedni sposób. Zdarzają się przypadki, kiedy niezwykle trudno jest określić poszukiwaną informację w sposób umożliwiający znalezienie jej przez wyszukiwarke.

## 1.1. Cele pracy

Celem poniższej pracy jest skonstruowanie chatbota, który będzie wyszukiwał informacje z określonej dziedziny wiedzy w zasobach polskojęzycznej Wikipedii. Wyszukiwanie będzie się odbywało na podstawie zapytania sformułowanego przez użytkownika w języku naturalnym. Jeśli znajdzie taka potrzeba chatbot powinien poprowadzić z użytkownikiem rozmowę, która pozwoli na ujednoznaczenie zapytania – dokładne określenie jakiego rodzaju informacja ma zostać wyszukana. Wynikiem działania ma być adres strony (lub krótka lista adresów), na której można znaleźć odpowiedź na zadane pytanie.

## 1.2. Zawartość pracy

W rozdziale 2 przedstawiono podstawowe informacje dotyczące zagadnień przetwarzania języka naturalnego, istniejących rozwiązań oraz sposobu działania Wikipedii.

W rozdziale 3 zostaje szczegółowo opisany, stworzony w ramach pracy system. W tym rozdziale znajduje się opis architektury rozwiązania – komponentów z jakich się składa, wykorzystanego oprogramowania, zastosowanych algorytmów oraz zagadnienia związane z przetwarzaniem języka naturalnego.

Rozdział 4 zawiera prezentację działania systemu na konkretnych przykładach, wraz z analizą otrzymanych wyników oraz porównaniem z innymi rozwiązaniami.

Podsumowanie pracy znajduje się w rozdziale 5 – są tam informacje o zrealizowanych funkcjonalnościach; napotkanych problemach; zadaniach, których nie udało się zrealizować; możliwościach dalszego rozwoju.

## 2. Wprowadzenie

W tym rozdziale przedstawiono podstawowe informacje na temat zagadnień, które mają związek z tematyką niniejszej pracy.

### 2.1. Przetwarzanie języka naturalnego

Przed przystąpieniem do opisywania zagadnienia jakim jest przetwarzanie języka naturalnego, należy najpierw zdefiniować pojęcie języka naturalnego.

Język naturalny to język powstały na drodze rozwoju historycznego, zróżnicowany geograficznie i społecznie, przeciwstawiający się z jednej strony językom sztucznym (jak np. esperanto), z drugiej zaś językom formalnym i językom programowania. Od języków sztucznych różni się polisemią swoich wyrażań oraz tym, że podlega ciągłym zmianom. [4]

Język naturalny jest dla człowieka podstawowym narzędziem reprezentacji oraz przechowywania wiedzy. Odgrywa także bardzo istotną rolę w komunikacji z innymi ludźmi. Reprezentacja wiedzy przy pomocy języka naturalnego jest zazwyczaj niestrukturalizowana – jest zrozumiała dla ludzi, natomiast sprawia problemy maszynom.

Przetwarzanie języka naturalnego (ang. natural language processing, NLP) to interdyscyplinarna dziedzina, łącząca zagadnienia sztucznej inteligencji i językoznawstwa, zajmująca się automatyzacją analizy, rozumienia, tłumaczenia i generowania języka naturalnego przez komputer. System generujący język naturalny przekształca informacje zapisane w bazie danych komputera na język łatwy do odczytania i zrozumienia przez człowieka. Zaś system rozumiejący język naturalny przekształca próbki języka naturalnego na bardziej formalne symbole, łatwiejsze do przetworzenia dla programów komputerowych. [14]

Aby umożliwić maszynom wykorzystywanie wiedzy stworzono formalne sposoby jej reprezentacji. Przykładem mogą być ontologie opisujące wybrane dziedziny wiedzy poprzez określenie zbioru pojęć oraz relacji między nimi. Taki sformalizowany sposób pozwala maszynie na przeprowadzenie operacji wnioskowania, jednak z drugiej strony jest dla człowieka nieintuicyjny, przez co ilość wiedzy tak reprezentowanej jest mała. Algorytmy przetwarzające język naturalny pozwalają programom komputerowym na operowanie ludzką wiedzą, która utrwalona została w najpopularniejszym „formacie”. Do przykładowych zastosowań NLP można zaliczyć:

- tłumaczenie automatyczne – tłumaczenie tekstu z jednego języka naturalnego na inny,
- rozstrzyganie autorstwa, wykrywanie plagiatów – określanie autora wybranego tekstu na podstawie stylu, słownictwa, porównywanie dokumentów,



- streszczanie – przekształcenie tekstu mające na celu redukcję jego objętości przy jednoczesnym zachowaniu najważniejszych faktów,
- korekta pisowni – wskazywanie błędów w tekście oraz sugerowanie odpowiednich poprawek,
- klasyfikacja dokumentów na podstawie ich tematu/treści,
- wyszukiwanie dokumentów,
- ekstrakcja informacji – analizowanie tekstu w celu uzyskania z niego informacji (faktów, danych), które niesie,
- odpowiadanie na pytania – generowanie odpowiedzi w języku naturalnym, w którym zostało zadane pytanie.

Z przetwarzaniem języka naturalnego wiąże się szereg problemów, które muszą zostać rozwiązane, aby umożliwić efektywną pracę programów komputerowych. Podstawowym z nich jest brak odpowiedniej precyzji na wszystkich poziomach języka:

- fonetyka, fonologia – identyczne brzmienie różnych słów (homofonia),
- morfologia – identyczny zapis różnych słów (homonimia), problem z identyfikacją części mowy,
- składnia – zdanie można rozebrać (dokonać analizy składniowej) na wiele sposobów, problem z identyfikacją części zdania,
- semantyka – znaczenie wypowiedzi lub słów zmienia się w zależności od kontekstu, w którym występują (polisemia),
- pragmatyka – zrozumienie wypowiedzi wymaga pewnej wiedzy o otaczającym świecie.

Sam opis języka naturalnego jest niezwykle skomplikowany. Nawet jeśli wziąć pod uwagę reguły gramatyczne, to sama liczba wyjątków od reguł może być przytłaczająca. Nie należy także zapominać o tym, że język będący w użyciu jest tworem żywym i ulega ciągłym zmianom – ewoluuje. W zależności od rozpatrywanego języka problematyka NLP może koncentrować się na innych zagadnieniach. Przykładem może być język angielski, który ma prostą fleksję i budowę zdania oraz język polski, w którym fleksja jest bardzo rozbudowana, a szyk zdania może być dowolny.

Można wyróżnić dwa podejścia w przetwarzaniu języka naturalnego:

- Podejście gramatyczne – lingwistyka formalna, definiowanie języka w sensie matematycznym poprzez zbiór reguł,
- Podejście statystyczne – określanie struktury i reguł języka poprzez analizowanie bardzo dużych ilości tekstu.

## 2.2. Wyszukiwarki internetowe

Wyszukiwarka internetowa jest to oprogramowanie, które pozwala na znalezienie dokumentów w sieci Internet zaspokajających potrzebę informacyjną użytkownika[1]. W budowie wyszukiwarki można wyróżnić trzy zasadnicze elementy[3].

Pierwszym z nich jest pajak (robot internetowy) – program przeglądający sieć Internet w celu pozyskania danych. Przeszukiwanie rozpoczyna się od zdefiniowanych punktów startowych (adresów wybranych stron internetowych). Dokumenty są pobierane, a następnie najczęściej wstępnie analizowane (ekstrakcja linków do dalszych poszukiwań) i umieszczane w tymczasowym magazynie danych. Bardziej zaawansowane mechanizmy mogą przeprowadzić ocenę strony przed jej pobraniem w celu określenia, czy jest w ogóle sens ją pobierać (czy znajdują się na niej cenne informacje). Czasami zdarza się, że właściciel serwisu nie życzy sobie indeksowania i umieszcza informacje o tym fakcie w pliku robots.txt (ang. *Robots Exclusion Protocol*), który powinien być respektowany przez pajaka. Przy konstruowaniu pajaka należy także pamiętać o odpowiednim limitowaniu pobieranych danych. Pomiędzy kolejnymi żądaniami powinien być pewien odstęp czasowy – wyszukiwarka nie może przeprowadzać ataku typu odmowa usługi (ang. *Denial of Service*) zużywając całe dostępne pasmo internetowe przeglądanego serwisu.

Kolejnym elementem wyszukiwarki jest mechanizm indeksujący strony. Indeks określa się wyspecjalizowaną bazę danych, która przechowuje informacje o dokumentach. Indeks powinien być zbudowany w taki sposób, aby możliwe było szybkie wyszukanie dokumentów, które spełniają odpowiednie kryteria. Istnieją gotowe mechanizmy wyszukiwania pełnotekstowego, które można wykorzystać podczas konstruowania indeksera – przykładem może być Apache Lucene lub Sphinx. Przed umieszczeniem w indeksie dokument musi zostać odpowiednio przetworzony[1][3]. Na samym początku należy oczyścić dokument (np. usunąć kod HTML, wyekstrahować tekst z pliku PDF). Później jest on poddawany procesowi tokenizacji – dzielenia tekstu na słowa. Te najbardziej pospolite (ang. *stop-words*) są zazwyczaj z tekstu usuwane. Kolejnym etapem jest przeprowadzenie analizy morfologicznej – jest to szczególnie istotny etap dla języków z rozbudowaną fleksją takich jak język polski. Wyrazy w języku naturalnym mogą mieć różną formę w zależności od kontekstu, aby umożliwić powiązanie wyrazów z danym leksemem trzeba w jakiś sposób charakteryzować klasy wyrazów. Stosuje się zazwyczaj jeden z poniższych sposobów:

- stemming – polega na określeniu rdzenia wyrazu. Stemmer nie zawsze musi zwrócić ciąg znaków będący prawidłowym rdzeniem, ważne jest aby dla wyrazów należących do danego leksemu rdzeń był zawsze taki sam.
- lematyzacja – polega na sprowadzeniu odmienionej formy wyrazu do tematu. W procesie lematyzacji istotny jest kontekst, w którym wyraz się znajduje.

Ostatnim elementem jest mechanizm odpowiadania na zapytania. Jego zadaniem jest pobranie zapytania od użytkownika, a następnie przetworzenie na kryteria, które pozwolą wyszukać odpowiednie dokumenty w indeksie. W tym procesie usuwa się słowa pospolite oraz takie, które mogłyby zakłócić proces wyszukiwania (np. poprawianie błędów ortograficznych, literówek – chociaż zdarzają się przypadki, kiedy to właśnie słowa z błędami są poszukiwane). Rezultatem wyszukiwania jest zazwyczaj lista dokumentów, która powinna być posortowana według trafności (lub np. kryteriów zdefiniowanych przez użytkownika). Interfejs użytkownika powinien być prosty i intuicyjny w obsłudze, a sam proces wyszukiwania możliwie krótki. Im dłużej użytkownik musi czekać na wynik, tym większe jest prawdopodobieństwo, że strona zostanie zamknięta przed wyświetleniem wyników. Dobrym rozwiązaniem jest implementacja pamięci podręcznej na ostatnie oraz popularne zapytania (proces wyszukiwania może być przeprowadzany okresowo, żeby wyniki zawsze były gotowe).

## 2.3. Chatbot

Chatbot lub chatterbot to program komputerowy mający za zadanie prowadzenie inteligentnych rozmów z ludźmi przy pomocy metod głosowych lub tekstowych. Odpowiednio skonstruowane programy redukują barierę komunikacyjną człowiek-komputer. Pozwalają człowiekowi na porozumienie się z maszyną w języku naturalnym.

Ważnym aspektem brany pod uwagę podczas tworzenia chatbotów jest ich zdolność do przekonania człowieka, że rozmawia z innym człowiekiem, a nie programem komputerowym. Alan Turing w 1950 roku zaproponował test (nazwany później jego nazwiskiem) sprawdzający zdolności maszyny do prowadzenia inteligentnej konwersacji[10].

### 2.3.1. Zastosowania

Początkowo jedynym zadaniem chatbotów było udawanie przed rozmówcą człowieka. Wraz z rozwojem Internetu zaczęto dostrzegać korzyści płynące z zastosowania chatbotów w biznesie. Mogą one służyć do celów promocyjno-rozrywkowych – przykładem takiego rozwiązania jest Fido<sup>1</sup>, robot który reklamuje produkty firmy Fido Interactive, jednocześnie umożliwiając porozmawianie na wiele tematów nie związanych z firmą.

Rosnąca popularność sklepów internetowych zaowocowała powstaniem różnego rodzaju asystentów, którzy zbierają i udzielają informacje. Wirtualny sprzedawca on-line pomoże klientowi z doбором produktów, poinformuje o aktualnych promocjach, przyjmie zamówienie na pizzę, czy zarejestruje reklamację. Istnieją także inteligentne systemy pomocy, gdzie zamiast przeszukiwać spisy treści, czy indeksy, można po prostu zadać pytanie.

Dla chatbotów znaleziono także nietypowe zastosowanie – przestępcy wykorzystują je masowego spamowania, podszywania się oraz wyłudzenia prywatnych informacji (np. numerów kont bankowych) za pośrednictwem portali społecznościowych, poczty elektronicznej oraz czatów.

### 2.3.2. Sposób działania

Jeden z pierwszych chatbotów była ELIZA, opracowywana przez Josepha Weizenbauma w latach 1964-1966 na uniwersytecie MIT. Poprzez tekstowy terminal użytkownik wprowadzał tekst, na który bot odpowiadał. Jego działanie opierało się na prostym dopasowywaniu wzorców. Po wpisaniu przez użytkownika zdania, program wyszukiwał słowa kluczowe, dla których miał wcześniej przygotowane odpowiedzi. Na początku sprawiało to wrażenie rozmowy z innym człowiekiem. Przykładowo na zdanie zawierające słowo *ojciec* program może odpowiedzieć „Napisz coś więcej o swojej rodzinie”.

Innym znanym chatbotem jest A.L.I.C.E. (Artificial Linguistic Internet Computer Entity), którego rozwój zapoczątkował Richard Wallace w 1995 roku. Na jego potrzeby został stworzony AIML (Artificial Intelligence Markup Language), dialekt XML'a. Umożliwia on na zdefiniowanie bazy wiedzy bota, która będzie w całości decydowała o sposobie prowadzenia konwersacji. Jednym z głównych założeń projektowych AIML była łatwość nauki i prostota. Poniżej zaprezentowana jest przykładowa reguła w tym języku:

```
<category>
  <pattern>MAM NA IMIĘ *</pattern>
  <template><star/> to bardzo ładne imię.
```

<sup>1</sup>Dostępny pod adresem: [www.fido.pl](http://www.fido.pl)

```

    Napisz coś więcej o sobie.</template>
</category>
<category>
  <pattern>NAZYWAM SIĘ *</pattern>
  <template>
    <srai>MAM NA IMIĘ <star/></srai>
  </template>
</category>

```

Można łatwo dostrzec, że takie rozwiązanie daje dobre rezultaty dla języka angielskiego. Natomiast przygotowanie zestawu reguł dla języka polskiego wymaga dużo więcej pracy. Luźny szyk zdania i rozbudowana fleksja wymaga dodania np. możliwości dopasowywania tekstu niezależnie od płci osoby, która go wprowadza. Taki cel można by osiągnąć przykładowo dodając obsługę wyrażeń regularnych.

## 2.4. Istniejące wyszukiwarki

W sieci Internet istnieje wiele serwisów wyszukiwawczych, którym można zadawać pytania w języku naturalnym. Niestety większość z nich obsługuje tylko język angielski, ze względu ilość treści publikowanej w tym języku oraz liczbę użytkowników posługujących się nim.

Jednym ze znanych rozwiązań jest serwis `answers.com`. Umożliwia on przeszukiwanie określonej grupy źródeł (słowniki, encyklopedie, przewodniki) oferując obok angielskiego, cztery inne wersje językowe: francuską, niemiecką, włoską i hiszpańską. Jako odpowiedzi wyświetlane są w całości lub we fragmentach artykuły traktujące na dany temat. Ciekawym rozwiązaniem jest część serwisu o nazwie WikiAnswers, która umożliwia zarejestrowanym użytkownikom udzielanie odpowiedzi na pytania oraz ich modyfikowanie. Kiedy odpowiedź nie może być znaleziona w głównych źródłach wyszukiwarki zwracane są wyniki z serwisu Google.

Innym znanym serwisem jest `ask.com`, który łączy typową wyszukiwarkę internetową z serwisem odpowiadającym na pytania. Wynikiem wyszukiwania jest fragment strony zawierający odpowiedź na zadane pytanie (o ile udało się taki znaleźć), a po nim tradycyjna lista stron wraz z krótkimi opisami.

Przykładem nietypowego rozwiązania jest WolframAlpha. Nie jest to wyszukiwarka internetowa, a obliczeniowa baza wiedzy (ang. *computational knowledge engine*)[17]. Dane nie są pozyskiwane w sposób automatyczny – są to najczęściej dane tablicowe odpowiednio wcześniej przygotowywane przez ludzi. Zapytania mogą być formułowane w języku naturalnym (obsługiwany jest jedynie język angielski) – przy czym to co odróżnia WolframAlpha od innych wyszukiwarek to możliwość wykonywania operacji na danych. Przykładowym zapytaniem może być: *What is the mean gross domestic product of Poland for last five years?* (ang. Jaka jest średnia wartość produktu krajowego brutto Polski w ciągu ostatnich pięciu lat?). Odpowiedzią na te pytanie jest obliczona wartość, tabele z danymi oraz wykresy.

Warto także wspomnieć o najpopularniejszej wyszukiwarce ogólnego przeznaczenia jaką jest Google. Jest to wyszukiwarka posiadająca największy indeks stron internetowych. Eksperymentalnie dla języka angielskiego wprowadzono funkcję odpowiadania na proste pytania. Przykładowo dla zapytań: *Where Barack Obama was born?* (ang. Kiedy urodził się Barack Obama?), *What is the capital city of Poland?* (ang. Co jest stolicą Polski?) zostały udzielone odpowiedzi: *Barack Obama born – August 4, 1961, Poland Capital city – Warsaw*. Jak widać na podanych przykładach, nie działa on idealnie. Zakres tematyczny takich poszukiwań jest też bardzo ograniczony - na pytanie typu *Who played Angus MacGyver?* zostanie przedstawiona normalna lista wyników.

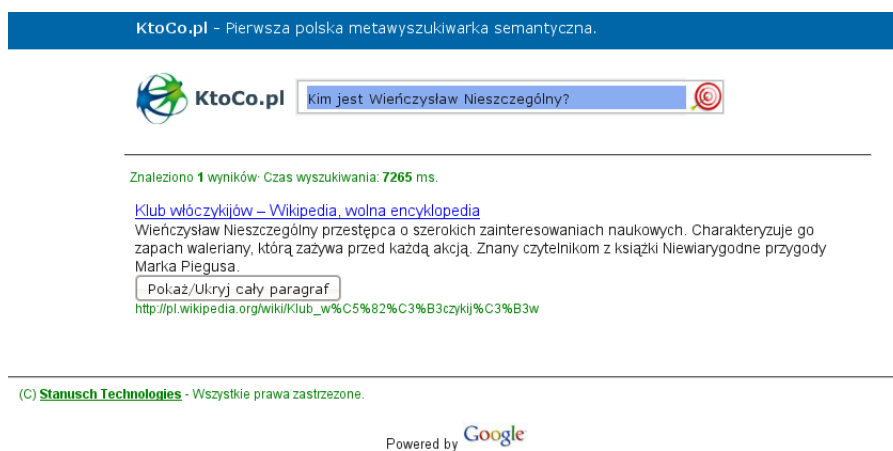
Poniżej zostały opisane dwie wyszukiwarki, które obsługują zapytania w języku polskim.

### 2.4.1. KtoCo.pl

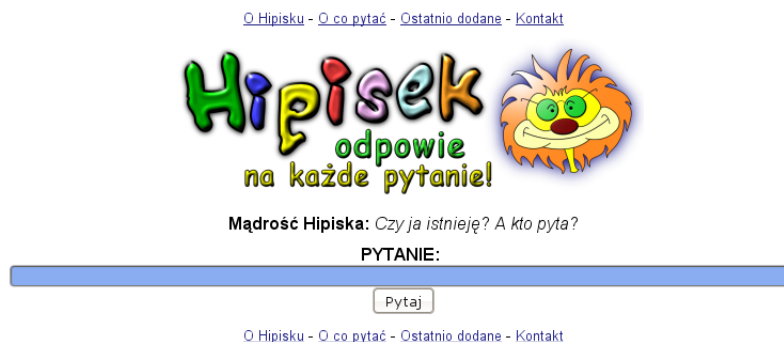
<http://KtoCo.pl> jest projektem badawczym firmy Stanusch Technologies sp. z o.o, który oferuje wyszukiwanie dokumentów oraz informacji w oparciu o sztuczną inteligencję[5]. System po przetworzeniu zapytania, które może być sformułowane w języku naturalnym, powinien wskazać odpowiedni fragment strony internetowej zawierający odpowiedź na zadane pytanie. Nie zawsze jednak się to udaje, a wyniki są czasem wielokrotnie powielane. Poza notką, że przy przetwarzaniu zapytania wykorzystywana jest wyszukiwarka Google, na stronie nie ma dokładniejszych informacji na temat sposobu działania tego mechanizmu.



Rysunek 2.1: Interfejs wyszukiwarki KtoCo.pl.



Rysunek 2.2: Interfejs wyszukiwarki KtoCo.pl, odpowiedź na zadane pytanie.



Rysunek 2.3: Interfejs wyszukiwarki Hipisek.pl.

### 2.4.2. Hipisek.pl

Marcin Walas w ramach swojej pracy magisterskiej opracował system odpowiadający na pytania, który dostępny jest pod adresem <http://hipisek.pl>[8]. Został on stworzony z myślą o obsłudze zapytań o miejsce i czas. Mechanizm wykorzystuje proste narzędzia lingwistyczne w celu wyszukania odpowiedniej informacji oraz wygenerowania odpowiedzi. Dostępna w Internecie aplikacja indeksuje strony dwóch serwisów: informacyjnego <http://www.tvn24.pl> oraz plotkarskiego <http://www.pudelek.pl>.

**Czy wykryto w Polsce wirusa grypy**

**Tematem pytania jest:** Polska [Polsce]  
**Akcja pytania:** wykryć [wykryto]

**Odpowiedź:**

Odpowiednie procedury zostały wdrożone w stosunku do ich współpasażerów - zapewnił Tomasz Misztal z Głównego Inspektoratu Sanitarnego.

**W Polsce wykryto już zatem w sumie 19 przypadków nowego szczepu grypy.**

Wirus A/H1N1 to mieszanka świńskich, ludzkich i ptasich wirusów grypy.

**Dane źródłowe:**  
 Źródło: <http://www.tvn24.pl/0,1607754,0,1,swinska-grypa-w-polsce-atakuje-mloдых.wiadomosc.html>  
 Data: 2009-07-01  
 Ocena: 0.69375

Rysunek 2.4: Interfejs wyszukiwarki Hipisek.pl, odpowiedź na zadane pytanie.

System składa się z dwóch zasadniczych elementów. Pierwszym z nich jest robot internetowy, który pobiera nowe treści z serwisów, analizuje je oraz indeksuje. Drugim jest mechanizm wyszukiwania, który działa w kilku etapach[7]:

1. Przyjęcie zapytania – serwis odbiera zapytanie od użytkownika oraz je weryfikuje (np. sprawdza liczbę zapytań z danego adresu IP).
2. Przetworzenie zapytania do reprezentacji formalnej.

3. Wyszukanie odpowiednich stron, które mogą być potencjalnym źródłem odpowiedzi. Do tego celu została wykorzystana wyszukiwarka pełnotekstowa Sphinx (<http://www.sphinxsearch.com/>), która jest rozpowszechniana jako wolne oprogramowanie.
4. Ekstrakcja danych oraz wygenerowanie odpowiedzi. Wraz z odpowiedzią dostarczany jest kontekst, adres źródła oraz datę pobrania.

## 2.5. Wikipedia, wolna encyklopedia

Wikipedia jest wielojęzycznym projektem internetowej encyklopedii, który udostępnia treść na wolnej licencji (ang. *free license*). Najbardziej charakterystycznym elementem tego projektu jest jego otwartość – każdy użytkownik, nawet anonimowy, może dodawać i modyfikować treść znajdującą się w serwisie. Według rankingu firmy Alexa Internet, Wikipedia znajduje się w pierwszej dziesiątce najczęściej odwiedzanych witryn internetowych na świecie.



Rysunek 2.5: Logo

### 2.5.1. Historia i rozwój

W marcu 2000 roku Jimmy Wales oficjalnie zakłada projekt internetowej encyklopedii o nazwie Nupedia, który jest finansowany przez firmę Bomis. Jego współpracownik Larry Sanger zostaje redaktorem naczelnym. W celu zapewnienia odpowiedniego poziomu jakości publikowanych haseł, zostaje zastosowany siedmiostopniowy proces redakcyjny. Główną rolę mieli odgrywać w nim eksperci, którzy mając udokumentowane wykształcenie z danej dziedziny biorą udział w pracy nad wersją finalną publikowanego dokumentu. Przerost biurokracji okazał się jednak przeszkodą w rozwoju projektu – w listopadzie 2000 roku Nupedia miała zaledwie dwa pełne artykuły[15].

W styczniu 2001 roku utworzono poboczny projekt o nazwie Wikipedia, który miał na celu umożliwienie wspólnego redagowania artykułów, zanim miały one trafić do sprawdzenia i publikacji w Nupedii. Brak biurokracji oraz otwartość projektu zaczęły stopniowo przyciągać coraz więcej edytorów. Pod koniec roku 2001 liczba artykułów wynosiła około 20000 dla 18 wersji językowych. Podczas gdy popularność Wikipedii rosła, zaangażowanie w Nupedię stopniowo malało, aż została ona zamknięta 23 września 2003 roku mając 24 ukończone hasła oraz 74 zaczęte, o różnym stopniu zaawansowania. 20 czerwca 2003 roku Jimmy Wales oficjalnie ogłasza założenie spółki non-profit – Wikimedia Foundation Inc. – organizacji powołanej do opieki nad Wikipedią i jej innymi projektami opartymi na idei wiki.

W styczniu 2010 anglojęzyczna wersja Wikipedii ma ponad 3 miliony haseł, a jej polskojęzyczny odpowiednik, założony 26 września 2001 została założona roku, prawie 700000. W sumie wszystkie edycje językowe mają ponad 14 milionów haseł[13].

### 2.5.2. Najważniejsze zasady

W celu przyspieszenia prac nad rozwojem Wikipedii, a także aby zapobiegać konfliktom wśród edytorów, powstał zbiór zasad, do którego każdy powinien się stosować. Od początkujących wikipedystów nie jest wymagana natychmiastowa znajomość wszystkich zasad. Istnieje jednak pięć podstawowych reguł, o których każdy powinien wiedzieć, zanim zacznie edytować[16].

- **Wikipedia jest encyklopedią.** To nie jest miejsce na reklamę, propagowanie swojego światopoglądu, blogiem, zbiorem odnośników, słownikiem, czy gazetą. Artykuły uznawane za nieency-

klopedyczne są kasowane, a kryteria encyklopedyczności są ustalane podczas dyskusji. Artykuły powinny być napisane w oparciu o wiarygodne, opublikowane źródła. Wikipedia nie jest miejscem na przedstawianie wyników własnych badań, teorii czy odkryć – wszystkie opisywane fakty powinny być dla czytelnika weryfikowalne. Edytorzy powinni dokładać starań, aby artykuły były jak najbardziej precyzyjne.

- **Artykuły są pisane w sposób neutralny.** Kiedy na dany temat istnieje kilka odmiennych poglądów, to zgodnie z zasadą neutralnego punktu widzenia należy przedstawić każdy z nich, bez wskazywania, który z nich jest słuszny. Należy wtedy szczególną uwagę przywiązać do źródeł, na których opiera się dany artykuł, tak aby każdy z punktów widzenia był poparty rzetelnymi publikacjami.
- **Wikipedia to wolny zbiór wiedzy**, który może być edytowany i wykorzystywany przez każdego. Treść jest publikowana na licencjach GNU Free Documentation License oraz CC-BY-SA 3.0<sup>2</sup>. Każdy edytujący musi przestrzegać prawa autorskiego oraz pamiętać o tym, że tworząc artykuł nie staje się jego właścicielem. Inne osoby na pewno będą później wprowadzać do niego poprawki.
- **Edytując przestrzegaj etykiety.** Wikipedyści powinni szanować innych edytorów, nawet jeśli się z nimi nie zgadzają. Podczas dyskusji należy skupiać się na problemie, a nie na dyskutantach. Nie należy wdawać się w wojny edycyjne – nie można forsować swoich wersji artykułów, należy w drodze konsensusu wypracować wspólną wersję.
- **Wikipedia nie ma sztywnych zasad**, za wyjątkiem tych pięciu wymienionych. Nie należy się obawiać, że edytując artykuł można coś popsuć – wszystko zostaje zapisane w historii edycji, poprzednie wersje artykułów nie są kasowane, dlatego każdy bałagan można szybko wycofać.

### 2.5.3. Społeczność edytorów oraz wiarygodność projektu

Wikipedia jest tworzona przez wolontariuszy, którzy w swoim wolnym czasie pracują nad artykułami. Nie ma żadnych ograniczeń jeśli chodzi o to, kto może edytować. Może się to wydawać na początku dziwne, że Wikipedia w ogóle funkcjonuje i stanowi wartościowe źródło wiedzy, skoro ktoś może, niczym uliczny wandal malujący na murze, wpisać np. wulgaryzmy.

Wikipedia posiada wiele mechanizmów, które pozwalają na przeciwdziałanie wandalizmom. Najważniejszym z nich jest strona specjalna *Ostatnie zmiany*<sup>3</sup>, na której wyświetlana jest lista ostatnich edycji w projekcie. Każdy artykuł posiada historię zmian – to co zostało skasowane, można jednym kliknięciem przywrócić. Ochotnicy (w tym także specjalne programy – boty) sprawdzają ostatnie edycje i wycofują dużą część wandalizmów zaraz po tym, jak zostały one wprowadzone.

Samo istnienie takiej strony nie rozwiązuje wszystkich problemów – zdarzają się okresy, kiedy nikt nie „patroluje” ostatnich edycji. W związku z tym w listopadzie 2008 w polskojęzycznej Wikipedii uruchomiono rozszerzenie o nazwie Wersje oznaczone (ang. *Flagged Revisions*). Edycje osób anonimowych i zarejestrowanych z krótkim stażem wymagają akceptacji doświadczonego edytora zanim zostaną pokazane szerszej publiczności. Dzięki temu rozwiązaniu liczba wandalizmów, jaka jest widoczna dla przeciętnego czytelnika, znacząco spadła.

<sup>2</sup>Creative Commons – uznanie autorstwa – na tych samych warunkach.

<sup>3</sup>[http://pl.wikipedia.org/wiki/Specjalna:Ostatnie\\_zmiany](http://pl.wikipedia.org/wiki/Specjalna:Ostatnie_zmiany)



### 2.5.4. Projekty siostrzane

Oprócz Wikipedii istnieją także inne projekty fundacji Wikimedia, które uzupełniają encyklopedie o inne, cenne materiały. Te projekty także są oparte na mechanizmie wiki, a ich treść jest udostępniana na wolnych licencjach[15].

- **Wikisłownik** (ang. *Wiktionary*) – projekt internetowego słownika on-line wszystkich języków. Stanowi on połączenie słownika synonimów, słownika etymologicznego, ortograficznego, fleksyjnego, frazeologicznego, słownika imion, nazw własnych, skrótów oraz przysłów.
- **Wikibooks** – jest to biblioteka wolnych książek oraz podręczników. Zawiera takie publikacje jak książka kucharska, kursy językowe czy podręcznik fizyki matematycznej.
- **Wikinews** – projekt serwisu informacyjnego, w którym każdy może być reporterem opisującym bieżące wydarzenia z kraju i ze świata.
- **Wikicytaty** (ang. *Wikiquote*) – skarbnica cytatów, sentencji i maksym pogrupowanych według autorów i tematów.
- **Wikiźródła** (ang. *Wikisource*) – wolne repozytorium tekstów źródłowych i ich tłumaczeń. Projekt ma na celu gromadzenie materiałów, które znajdują się w domenie publicznej lub są dostępne na wolnej licencji. Uczestnicy tego projektu zajmują się między innymi digitalizacją publikacji.
- **Wikiversity** – to projekt poświęcony materiałom szkoleniowym oraz badaniom. Można tam znaleźć profesjonalne kursy jak i nieformalne samouczki. Ten projekt nie posiada polskiej wersji językowej.
- **Wikimedia Commons** – wielojęzyczne repozytorium multimediów. W tym projekcie przechowywane są pliki – grafiki, zdjęcia, nagrania oraz animacje, które można osadzać we wszystkich projektach fundacji.
- **Wikispecies** – wielojęzyczny katalog gatunków. „Gromadzimy dane o zwierzętach, roślinach, grzybach, bakteriach, archeonach, protistach oraz wszelkich innych formach życia.” – to cytat z polskojęzycznej strony głównej tego projektu.

### 2.5.5. Oprogramowanie

Wikipedia działa w oparciu o oprogramowanie MediaWiki, które specjalnie zostało stworzone dla tego projektu. Jest ono dostępne na wolnej licencji – GNU General Public License (GPL). Oprogramowanie to jest stale rozwijane, zostało napisane w języku PHP i wykorzystuje bazę danych MySQL. Dla programistów jest dostępne rozbudowane API, które pozwala na łatwe pobieranie danych i edytowanie stron. Istnieje także duża baza rozszerzeń, którymi można dostosowywać instalacje do swoich potrzeb.

Jednym z rozszerzeń, o którym warto wspomnieć w kontekście tej pracy to Semantic MediaWiki. Standardowa instalacja MediaWiki umożliwia wykorzystywanie szablonów do prezentacji informacji na temat różnych bytów w ten sam sposób. Argumentami szablonu są pary, nazwa pola oraz jego wartość. Poza jednolitym sposobem prezentacji, tych danych nie da się wykorzystać w sposób automatyczny. Ten problem rozwiązuje rozszerzenie Semantic MediaWiki, które pozwala na klasyfikowanie treści, a następnie na jej przetwarzanie. W kodzie artykułów można przy pomocy tagów oznaczać informacje oraz relacje z innymi artykułami. Dodatkowo rozszerzenie pozwala na import ontologii OWL DL do wiki, a

także eksport informacji w formacie OWL/RDF. Mając do dyspozycji odpowiednio przygotowane artykuły można eliminować duplikowanie informacji, a przez to zmniejszyć ilość błędów oraz uspołnić prezentowane informacje (np. poprzez zastąpienie ręcznie tworzonych zestawień, generowanymi automatycznie). Niestety Semantic MediaWiki ma znaczący wpływ na wydajność oprogramowania, dlatego nie zostało ono dotychczas włączone dla Wikipedii.

## 3. Opis stworzonego rozwiązania

Niniejszy rozdział zawiera opis aplikacji, która została stworzona na potrzeby tej pracy. W rozdziale 3.1 zostały krótko przedstawione główne założenia, na których opiera się realizacja projektu. Następnie rozdział 3.2 opisuje ogólnie budowę stworzonego rozwiązania. Rozdział 3.3 zawiera szczegóły dotyczące implementacji – zastosowanych technologii oraz bibliotek, konfiguracji środowiska. Sposób działania narzędzi do analizy słownictwa został zaprezentowany w rozdziale 3.4. Opis algorytmów działania wirtualnego rozmówcy został zawarty w sekcji 3.5. Rozdział 3.6 przedstawia mechanizm wyszukiwania informacji. Na końcu rozdział 3.7 opisuje szczegółowo przepływ danych oraz sterowania w systemie.

### 3.1. Założenia projektowe

Na końcowy kształt aplikacji największy wpływ miały założenia projektowe, od których zależały decyzje podejmowane na wszystkich etapach jej tworzenia.

- Rozwiązanie wykorzystuje tylko oprogramowanie i materiały dostępne na zasadach wolnych licencji.
- Aplikacja posiada budowę modułową – składa się z komponentów, które współpracują ze sobą poprzez ustalone interfejsy.
- Przeszukiwanym obszarem są zasoby Wikipedii, które koncentrują się na opisywaniu biografii oraz miejsc.
- Wysyłany do chatbota tekst nie może zawierać literówek. Zdania muszą być napisane z użyciem poprawnej polszczyzny.
- Chatbot oczekuje już w pierwszym zapytaniu informacji pozwalających na rozpoczęcie poszukiwań. Sytuacją idealną jest możliwość wskazania strony już po pierwszym zapytaniu.

Dzięki użyciu tylko wolnych zasobów, stworzone rozwiązanie może być bez przeszkód dalej rozwijane oraz wykorzystywane w celach praktycznych – np. jako dodatkowa wyszukiwarka dla Wikipedii, a nie tylko na potrzeby niniejszej pracy. Modularność aplikacji zwiększa znacząco elastyczność rozwiązania – pozwala na łatwe modyfikowanie i skalowanie systemu. Poszczególne komponenty mogą być wymieniane lub zwielokrotniane i przenoszone na inne maszyny.

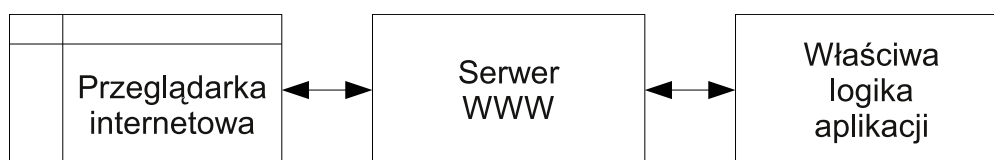
Uproszczenie wejścia dla chatbota jakim jest założenie poprawności wprowadzanych zdań wynika z tego, że celem tej pracy jest wyszukiwanie informacji, a nie automatyczna korekta tekstu, co samo w sobie może być tematem na odrębną pracę.

## 3.2. Architektura systemu

Opracowane rozwiązanie jest aplikacją internetową, zatem można je podzielić na dwie, zasadnicze części.

- Część kliencką, która jest wykonywana przez przeglądarkę internetową użytkownika. Jej głównym zadaniem jest prezentowanie informacji w przystępny sposób oraz dostarczenie interfejsu, który umożliwi pobranie zapytań.
- Część usługową, która jest wykonywana na dedykowanej maszynie. W niej znajduje się właściwa logika aplikacji, której zadaniem jest obsługa sesji użytkowników oraz przeprowadzanie procesu wyszukiwania informacji.

Obie te części łączy ze sobą serwer stron internetowych (rys. 3.1). Do jego zadań należy dostarczanie statycznych treści przeglądarce internetowej, takich jak: grafik, arkuszy stylów, skryptów JavaScript. Innym ważnym zadaniem jest uruchamianie aplikacji generujących treść w sposób dynamiczny – przekazuje komunikaty pomiędzy przeglądarką, a aplikacjami usługowymi. Serwer WWW może być częścią samej aplikacji, jednak zazwyczaj stanowi on odrębne oprogramowanie.

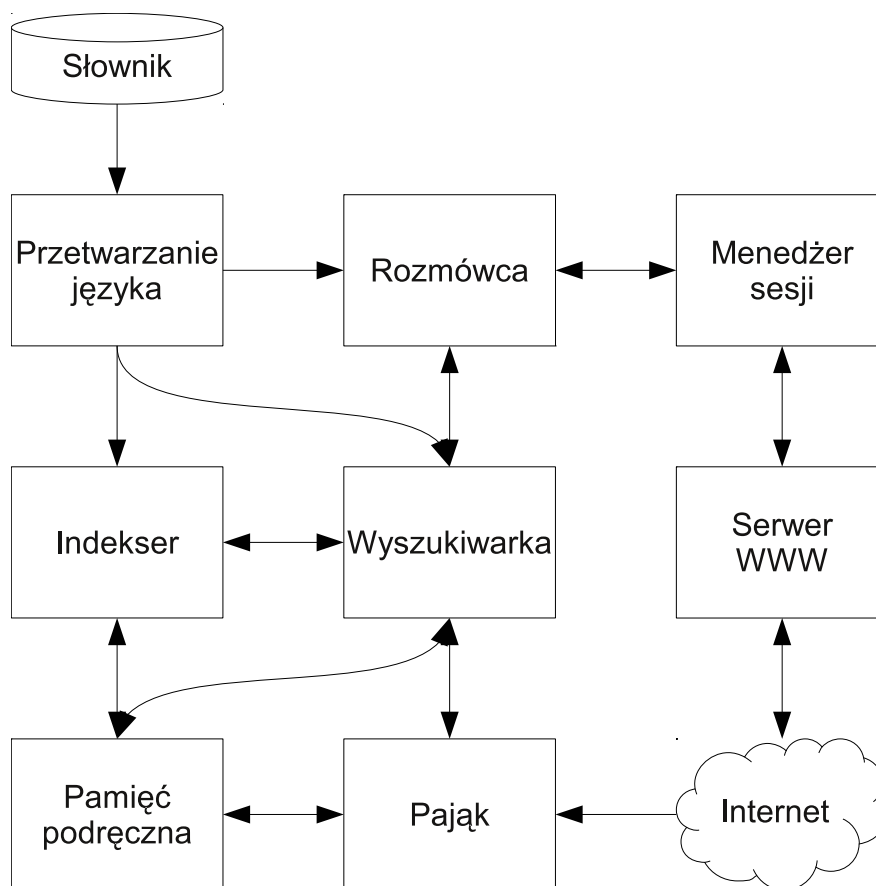


Rysunek 3.1: Środowisko wykonania aplikacji internetowej.

Cała aplikacja została podzielona na szereg logicznych modułów, z których każdy pełni odrębną funkcję. Na rysunku 3.2 zostały przedstawione zależności pomiędzy nimi.

Zadania modułów są następujące:

- **Serwer WWW** – przekazuje zapytania do menedżera sesji, a następnie zwraca odpowiedź przeglądarce klienta,
- **Menedżer sesji** – odpowiada za kontrolowanie sesji: tworzenie oraz niszczenie procesów, które zajmują się obsługą użytkowników, a także przekazuje do nich zapytania,
- **Rozmówca** – prowadzi rozmowę – analizuje tekst wpisywany przez użytkownika, wysyła zadania do wyszukiwarki oraz generuje odpowiedzi,
- **Przetwarzanie języka** – dostarcza pozostałym modułom API, które umożliwia analizowanie tekstu zapisanego w języku polskim, korzysta przy tym z zewnętrznych źródeł danych,
- **Wyszukiwarka** – koordynuje proces wyszukiwania, wysyła zadania do pająka oraz generuje listę wyników dla danego zapytania,
- **Pająk** – pobiera i wstępnie analizuje strony internetowe, które następnie przekazywane są do indeksacji,
- **Indekser** – analizuje pobrane strony pod kątem zawartej w nich treści oraz w odpowiedni sposób kataloguje,



Rysunek 3.2: Logiczny podział stworzonej aplikacji.

- **Pamięć podręczna** – dostarcza pozostałym modułom API, które umożliwi przechowywanie często wykorzystywanych informacji w pamięci o szybkim dostępie.

Jasne zdefiniowanie zadań dla każdego modułu ułatwia zaprojektowanie interfejsów, które będą określały zbiór realizowanych funkcji w sposób niezależny od ich implementacji.

### 3.3. Implementacja

Do zaimplementowania aplikacji został wybrany interpretowany język programowania Perl, który dobrze się sprawdza podczas pracy z danymi tekstowymi. Perl jest wolnym oprogramowaniem, które można znaleźć w większości instalacji systemów Unix i pochodnych. Jedną z zalet Perla jest ogromna biblioteka modułów – CPAN (ang. *Comprehensive Perl Archive Network*), które w większości są udostępniane na tych samych warunkach co Perl[9].

Stworzona aplikacja pracuje pod kontrolą systemu operacyjnego GNU/Linux, jednak ponieważ Perl jest przenośny, to po niewielkich modyfikacjach oprogramowanie powinno działać także na innych platformach. Na środowisko testowe została wybrana dystrybucja Linuksa o nazwie Gentoo, której używam na co dzień podczas pracy z komputerem. Cechą charakterystyczną tej dystrybucji jest elastyczność – wszystkie pakiety są kompilowane na komputerze użytkownika, co pozwala na dokładne dopasowanie oprogramowania do wymagań użytkownika i sprzętu, na którym ma pracować.

Główny interfejs użytkownika to statyczna strona HTML, której wygląd jest wzorowany na skórcie Vector oprogramowania MediaWiki. Wysyłanie zapytań odbywa się z wykorzystaniem technologii

AJAX (ang. *Asynchronous JavaScript and XML*), dzięki czemu interakcja użytkownika z serwerem nie wymaga ciągłego przeładowywania całego dokumentu. W tym celu została użyta biblioteka jQuery JavaScript Library w wersji 1.4.2, która udostępnia szereg narzędzi ułatwiających tworzenie dynamicznych stron WWW. Dodatkowo skrypt na stronie okresowo odpytuje serwer, czy nie pojawiły się nowe wiadomości, które należałoby wyświetlić użytkownikowi (np. wyniki czasochłonnego wyszukiwania).

Do serwowania stron internetowych wykorzystane zostało najpopularniejsze oprogramowanie tego typu – Apache HTTP Server<sup>1</sup>. Poniżej jest przedstawiona wykorzystywana konfiguracja serwera, która zakłada, że aplikacja znajduje się w katalogu `/var/www/localhost`.

```
<VirtualHost *:80>
  ServerName localhost
  ServerAdmin root@localhost

  DocumentRoot "/var/www/localhost/htdocs"

  <Directory "/var/www/localhost/htdocs">
    Options Indexes FollowSymLinks ExecCGI
    AddHandler cgi-script .pl
  </Directory>
</VirtualHost>
```

Przy tworzeniu dynamicznej części strony został wykorzystany wzorzec projektowy kontroler fasady (ang. *front controller*) – wszystkie żądania kierowane do serwisu są obsługiwane w jednym punkcie. W opracowanym rozwiązaniu jest nim skrypt CGI, który wstępnie sprawdza poprawność danych wejściowych, a następnie kieruje żądanie do menedżera sesji.

Menedżer sesji po otrzymaniu żądania sprawdza, czy istnieje dedykowany proces roboczy do obsługi danego użytkownika. Jeżeli takiego procesu nie ma to zostaje on utworzony, o ile na serwerze są dostępne zasoby do przeprowadzenia takiej operacji. Każdy proces roboczy otwiera gniazdo Uniksowe (ang. *Unix domain socket*), z którym skojarzony jest identyfikator sesji użytkownika. Na podstawie identyfikatora menedżer kontaktuje się z odpowiednim procesem wysyłając datagram zawierający żądanie. Przez określony czas skrypt CGI czeka na datagram z odpowiedzią od procesu roboczego, po jego przekroczeniu przeglądarka jest informowana o tym fakcie.

Odpowiedź aplikacji jest przesyłana do przeglądarki w formacie JSON (ang. *JavaScript Object Notation*). Dane zapisane w tym formacie mogą być w łatwy sposób przetwarzane przez skrypty.

Proces roboczy jest elementem aplikacji, który wykonuje właściwą pracę – zajmuje się wyszukiwaniem i obsługą zapytań użytkownika. Działa w oparciu o model zdarzeniowy – akcje są wykonywane asynchronicznie po zajściu konkretnego zdarzenia. Przykładami takich zdarzeń mogą być wysłanie wiadomości przez użytkownika lub pobranie strony internetowej.

Na wydajność całego systemu duży wpływ ma dostęp do pamięci podręcznej, która przechowuje często wykorzystywane dane. Jeśli potrzebne informacje znajdują się w pamięci podręcznej, nie trzeba angażować do ich uzyskania zewnętrznych, przeważnie wolniejszych źródeł danych. Dzięki takiemu podejściu kluczowe elementy systemu takie jak baza danych są odciążane, co pozwala na zwiększenie liczby obsługiwanych klientów. Jednym z popularnych systemów buforowania jest Memcached, którego API daje

---

<sup>1</sup>Na podstawie wyników badań firmy Netcraft – [http://news.netcraft.com/archives/2010/02/22/february\\_2010\\_web\\_server\\_survey.html](http://news.netcraft.com/archives/2010/02/22/february_2010_web_server_survey.html)

możliwość używania rozproszonych serwerów jako jednej, bardzo dużej tablicy asocjacyjnej. Zainstalowanie serwerów Memcached na maszynach, które mają niskie zużycie pamięci (np. dedykowanych serwerach WWW) pozwala ograniczyć marnotrawstwo oraz efektywnie zwiększyć sprawność całego systemu bez dokładania nowego sprzętu.

## 3.4. Słownik

Słownik jest jednym z ważniejszych komponentów aplikacji. Wykorzystywany jest podczas indeksowania i wyszukiwania dokumentów oraz prowadzenia rozmowy z użytkownikiem. Opracowana aplikacja dysponuje dwoma słownikami. Pierwszy z nich, opisany w rozdziale 3.4.1 zawiera informacje o odmianie wyrazów w języku polskim. Drugim jest słownik wyrazów oraz fraz bliskoznacznych, którego opis znajduje się w sekcji 3.4.2.

### 3.4.1. Formy fleksyjne

Ogólnodostępnych publikacji opisujących język polski jest niewiele. Podczas poszukiwań natrafiłem na trzy, które mogłyby zostać wykorzystane w aplikacji:

- Korpus IPI PAN – anotowany morfosyntaktycznie korpus języka polskiego, stworzony przez Zespół Inżynierii Lingwistycznej w Instytucie Podstaw Informatyki PAN, dostępny pod adresem <http://www.korpus.pl>,
- Słownik SJP.pl – słownik do gier słownych, tworzony w oparciu o mechanizm wiki, dostępny pod adresem <http://www.sjp.pl/>,
- Morfologik – projekt tworzenia polskich słowników morfosyntaktycznych, dostępny pod adresem <http://sourceforge.net/projects/morfologik/>.

Po zapoznaniu się z umową licencyjną korpusu IPI PAN, okazało się, że nie może on zostać wykorzystany w pracy. Udostępniane materiały nie są *wolne* – nie można ich w żaden sposób modyfikować, a ich używanie jest możliwe wyłącznie za pomocą dostarczonego oprogramowania – poliqrp.

Słownik SJP.pl został początkowo opracowany na potrzeby serwisu internetowego [kurnik.pl](http://kurnik.pl) z myślą o grach słownych. Jest on obecnie rozwijany przez internautów. Zasoby udostępnione do pobrania można podzielić na trzy kategorie: słowniki do sprawdzania pisowni, słowniki do gier oraz słownik odmian. Do celów pracy najbardziej przydatny jest słownik odmian zawierający wyrazy podstawowe skojarzone z formami pochodnymi, który jest dostępny na trzech wolnych licencjach: GPL, LGPL oraz CC-SA.

Morfologik to słownik do znakowania morfosyntaktycznego i syntezy gramatycznej. Został on opracowany przez Marcina Miłkowskiego przy wykorzystaniu zasobów słownika alternatywnego SJP.pl i udostępniony na takich samych warunkach. W Morfologiku każdej parze słów forma pochodna – forma bazowa towarzyszą znaczniki morfosyntaktyczne, które określają relację między słowami. To właśnie obecność tych informacji zadecydowała o wyborze Morfologika.

Same dane słownikowe to jednak nie wszystko, potrzebny jest mechanizm, który w szybki sposób będzie umożliwiał pobieranie informacji na temat słów. Perl ma wbudowaną obsługę tablic asocjacyjnych, więc pierwsze, naiwne podejście opierało się na wczytaniu całego słownika do pojedynczej tablicy. Operacja wczytywania danych trwała dość długo (około minutę), a wynikowy proces zajmował ponad 3GB pamięci operacyjnej. Najprostsza implementacja okazała się bardzo kosztowna. Wynika to stąd,

że interpreter Perla alokuje pamięć z bardzo dużym narzutem. Można się czasem spotkać z powiedzeniem, że aby przewidzieć zużycie pamięci przez program napisany w Perlu należy określić wymagania dla sensownej implementacji danego algorytmu, a następnie pomnożyć je przez 10, po czym może się okazać, że to i tak będzie za mało<sup>2</sup>.

Kolejną próbą było wykorzystanie biblioteki CDB::TinyCDB. Pozwala ona na tworzenie stałej bazy danych (zbioru nie da się modyfikować, aby coś zmienić trzeba go w całości przebudować), która działa jak szybka tablica asocjacyjna. Biblioteka oferuje dwa sposoby odczytu danych. Pierwszy z nich to otwarcie pliku z danymi i czytanie bezpośrednio z niego – rozwiązanie to sprawdza się dla pojedynczych zapytań lub kiedy plik z danymi jest wielki. Drugi sposób to odwzorowanie pliku w przestrzeni adresowej procesu (wywołanie systemowe `mmap` w systemach Uniksowych lub funkcja `API MapViewOfFile` w systemach Windows) pozwalające traktowanie pliku jak tablicy bajtów. To rozwiązanie ma dwie ważne zalety, pozwala na współdzielenie pamięci pomiędzy wszystkimi procesami mapującymi ten sam plik oraz nie wczytuje całości pliku od razu. Jeżeli czytany obszar nie znajduje się w pamięci, system operacyjny załaduje go z dysku (leniwe ładowanie, ang. *lazy load*). Niestety interfejs udostępniany przez omawianą bibliotekę nie daje możliwości efektywnego przechowywania słownika, dlatego rozmiar pliku wynikowego przekraczał 512MB.

Na koniec spróbowałem samodzielnie opracować strukturę danych, która przechowywałaby wszystkie potrzebne informacje nie marnując przy tym pamięci i pozwalając na szybki dostęp do nich. Po kilku eksperymentach powstało rozwiązanie napisane w języku C++, które wszystkie słowa organizowało w jedną, dużą posortowaną listę. Na początku pliku z danymi została umieszczona 32-bitowa liczba określająca wielkość słownika, następnie po niej umieszczone zostały 32-bitowe liczby wskazujące na pozycję każdego słowa w pliku. Słowo to łańcuch znaków zakodowany w systemie UTF-8, zakończony bajtem o kodzie 0, po nim występuje jeden bajt określający liczbę odnośników do innych słów. Relacja jest reprezentowana przez dwie liczby 32-bitowe, pierwsza wskazuje na pozycję słowa, a druga opisuje rodzaj relacji. Dzięki takiemu formatowi pliku rozmiar słownika udało zredukować się do 164MB. Dostęp do danych jest realizowany tak jak w przypadku CDB::TinyCDB przez odwzorowanie zawartości pliku w przestrzeni adresowej procesu. Do przetwarzania słownika stworzyłem dwie implementacje, które mają taki sam interfejs. Pierwsza z nich została napisana tylko w Perlu. Druga to klasa w języku C++, która przy pomocy interfejsu XS została „sklejona” z Perlem – jego kod może tworzyć i wykorzystywać obiekty tej klasy. Obie implementacje do przeszukiwania uporządkowanej listy wyrazów stosują wyszukiwanie binarne. Dla 3,5 miliona słów, które zawiera Morfologik potrzeba wykonać maksymalnie 22 operacje porównania, aby określić położenie wyrazu lub stwierdzić, że słownik go nie zawiera.

Warto w tym miejscu wspomnieć także o automatach skończonych (ang. *finite state machine*), które mogą być wykorzystane do zaimplementowania słownika o jeszcze większym stopniu upakowania danych. Dla języka polskiego istnieje implementacja opracowana przez Jana Daciuka, która została opublikowana na stronie [http://www.eti.pg.gda.pl/katedry/kiw/pracownicy/Jan.Daciuk/personal/fsa\\_polski.html](http://www.eti.pg.gda.pl/katedry/kiw/pracownicy/Jan.Daciuk/personal/fsa_polski.html). Niestety z powodu niejasnych zasad licencjonowania, wyżej wymienione oprogramowanie nie zostało wykorzystane w niniejszej pracy.

### 3.4.2. Synonimy

Opis danej rzeczy można wyrażać słowami na różne sposoby. Uwzględnianie wszystkich możliwości podczas poszukiwania jest dość pracochłonne. Wyszukiwarka powinna zatem dysponować me-

<sup>2</sup>Guts of Perl debugging: Debugging Perl memory usage – <http://perldoc.perl.org/perldebbugs.html>



chanizmem, który pozwoli na odnalezienie informacji pomimo odmiennych reprezentacji. Pierwszym krokiem do stworzenia takiego rozwiązania jest słownik synonimów.

Jako źródło materiału językowego został wykorzystany wolnodostępny słownik wyrazów bliskoznacznych języka polskiego znajdujący się pod adresem <http://synonimy.ux.pl/>. Jest to kolejny projekt Marcina Miłkowskiego. Słownik jest redagowany w oparciu o mechanizm wiki, gdzie każdy może się zarejestrować i brać czynny udział w tworzeniu. Jego zawartość jest udostępniana w różnych formatach na licencji GNU GPL. W niniejszej pracy została wykorzystana tekstowa wersja z 13 maja 2008 roku, zawierająca 13180 grup znaczeniowych.

Implementacja mechanizmu przeszukującego bazuje na tych samych pomysłach, co słownik form fleksyjnych opisany w poprzednim rozdziale (3.4.1). Indeks zawiera pojedyncze słowa lub frazy w oryginalnej formie, które skojarzone zostały z grupami znaczeniowymi. Można go przeszukiwać na dwa sposoby, używając całego klucza lub tylko prefiksu. Ta druga metoda pozwala na łatwe wyszukiwanie synonimów w tekście poprzez sprawdzanie tylko jednego słowa, a następnie próbowanie dopasowanie w całości odnalezionych fraz.

## 3.5. Chatbot

Chatbot realizuje jedną z dwóch kluczowych funkcji opracowywanego rozwiązania – prowadzenie rozmowy z użytkownikiem. Najważniejszym jego zadaniem jest pobranie od swojego rozmówcy kryteriów wyszukiwania wyrażonych w języku naturalnym, które pozwolą na znalezienie strony zaspokajającej jego potrzebę informacyjną. Stworzone rozwiązanie nie jest ograniczone do rozmów składających się tylko z dwóch wypowiedzi: pytania użytkownika i odpowiedzi systemu. Zaimplementowana została obsługa podstawowych pytań typu: kim jesteś?, co porabiasz? oraz standardowe elementy rozmowy takie jak przywitanie, podziękowania, pożegnanie. Chatbot dodatkowo reaguje na wypowiedzi niekulturalne, obraźliwe lub zawierające wulgaryzmy.

W kolejnych rozdziałach zostały szczegółowo opisane wybrane zagadnienia związane z chatbotem. W rozdziale 3.5.1 został opisany mechanizm odpowiedzialny za modelowanie dialogu. Następnie w rozdziale 3.5.2 zostały przedstawione metody analizy wypowiedzi człowieka. Sposób dopasowywania wzorców jest opisany w sekcji 3.5.3. Rozdział 3.5.4 zawiera informacje na temat generowania odpowiedzi. Informacje na temat interfejsu użytkownika, przy pomocy którego jest prowadzona rozmowa zostały zawarte w rozdziale 4.

### 3.5.1. Modelowanie dialogu

Ważnym aspektem chatbota jest umiejętność odpowiedniego reagowania na wypowiedzi człowieka. Wirtualny rozmówca powinien sprawiać wrażenie inteligentnego, dlatego każda jego reakcja musi przede wszystkim uwzględniać kontekst wypowiedzi. Przykładem komunikatu, którego interpretacja jest trudna bez posiadania informacji na temat wcześniejszego przebiegu rozmowy może być: „Tak, proszę.”.

Potrzebne jest zatem narzędzie, które umożliwiłoby modelowanie dialogu z uwzględnieniem kontekstu wypowiedzi. Aktualny temat rozmowy można określić jako stan, w którym system się znajduje. Natomiast odpowiedź jest funkcją bieżącego stanu i wypowiedzi rozmówcy. Taki przepływ sterowania można bardzo łatwo zamodelować wykorzystując koncepcję automatu skończenia stanowego (ang. *finite-state automaton*). Formalne definicje są następujące[2][12]:

- Deterministycznym automatem skończenie stanowym (ang. *deterministic finite state machine*) nazywamy uporządkowaną piątkę  $(\Sigma, S, s_0, \delta, F)$ , gdzie:
  - $\Sigma$  to alfabet wejściowy (skończony, niepusty zbiór symboli)
  - $S$  jest skończonym, niepustym zbiorem stanów
  - $s_0$  jest stanem początkowym, należy do zbioru  $S$
  - $\delta$  to funkcja przejścia stanów  $\delta : S \times \Sigma \rightarrow S$ , dla niedeterministycznego automatu skończonego jest to  $\delta : S \times \Sigma \rightarrow \mathcal{P}(S)$ , gdzie  $\delta$  zwraca zbiór stanów
  - $F$  to zbiór stanów końcowych, jest podzbiorem  $S$

W deterministycznych i niedeterministycznych automatach skończonych funkcja  $\delta$  nie musi być określona dla wszystkich par  $(q, x)$ , gdzie  $q \in S$  oraz  $x \in \Sigma$ . Kiedy automat znajduje się w stanie  $q$  i następnym symbolem jest  $x$ , a  $\delta(q, x)$  nie przyjmuje wartości, to powinien zostać zgłoszony błąd.

- Przetwornikiem skończenie stanowym (ang. *finite state transducer*) nazywamy uporządkowaną szóstkę  $(\Sigma, \Gamma, S, s_0, \delta, \omega)$ , gdzie:
  - $\Sigma$  to alfabet wejściowy (skończony, niepusty zbiór symboli)
  - $\Gamma$  to alfabet wyjściowy (skończony, niepusty zbiór symboli)
  - $S$  jest skończonym, niepustym zbiorem stanów
  - $s_0$  jest stanem początkowym, należy do zbioru  $S$
  - $\delta$  to funkcja przejścia stanów  $\delta : S \times \Sigma \rightarrow S$ , dla niedeterministycznego automatu skończonego jest to  $\delta : S \times \Sigma \rightarrow \mathcal{P}(S)$ , gdzie  $\delta$  zwraca zbiór stanów
  - $\omega$  to funkcja wyjścia

Jeżeli funkcja wyjścia odwzorowuje na symbol wyjściowy stan i symbol wejściowy ( $\omega : S \times \Sigma \rightarrow \Gamma$ ) to definicja odpowiada modelowi Mealy’ego. Natomiast jeżeli zależy wyłącznie od stanu ( $\omega : S \rightarrow \Gamma$ ) to definicja odpowiada modelowi Moore’a.

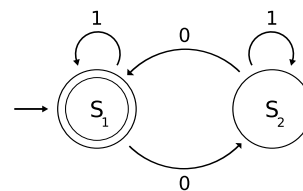
Automaty skończenie stanowe można reprezentować wykorzystując m.in. diagramy stanów (grafy skierowane pokazujące przepływ sterowania) lub też tablice przejść stanów.

Na potrzeby pracy definicja automatu skończonego została lekko zmieniona i rozszerzona o dodatkowe elementy, dzięki czemu chatbot może lepiej spełniać swoje zadanie. Implementacja maszyny stanowej modelującej dialog chatbota składa się z elementów podzielonych na trzy klasy:

- Kontroler – jest odpowiedzialny za sterowanie maszyną poprzez przyjmowanie symboli, sprawdzanie tranzycji oraz przełączanie stanów. Na każdą maszynę przypada tylko jeden obiekt tej klasy.
- Stan – określa aktualne położenie maszyny na grafie przepływu sterowania. Ze stanem można skojarzyć akcję, która ma zostać wywołana przez kontroler w chwili przejścia do niego. Przykładem może być stan pożegnania, po którego osiągnięciu użytkownikowi zostaje wyświetlony komunikat „Do rychłego zobaczenia!”. Każdy stan posiada także uporządkowaną listę tranzycji, które określają warunki przejścia do kolejnych stanów.
- Tranzycja – jest to funkcja przejścia pomiędzy stanami, określa kiedy może nastąpić zmiana bieżącego stanu.

Kontroler dysponuje dodatkowo historią stanów, która jest przechowywana w cyklicznym buforze o określonym rozmiarze. Przejście do nowego stanu powoduje automatyczne dodanie go do historii. Kiedy bufor jest pełny najstarszy stan zostaje nadpisany.

Dostarczenie symbolu na wejście maszyny powoduje podjęcie przez kontrolera próby przełączenia maszyny w kolejny stan. W tym celu kontroler rozpoczyna przeglądanie listy tranzycji, które są skojarzone z bieżącym stanem. Kolejność przeglądania jest zawsze ta sama, ponieważ zostaje ona ściśle określona podczas budowania relacji między stanami. Kiedy tranzycja akceptuje symbol, kontroler dokonuje przełączenia stanów. Odbywa się to poprzez dodanie do historii nowego stanu, a następnie wykonanie przypisanej do niego akcji.



Rysunek 3.3: Przykładowy diagram stanów

Kiedy żadna z tranzycji nie akceptuje symbolu, kontroler rozpoczyna przeglądanie historii. Kolejno od ostatniego do najwcześniejszego stanu sprawdzane są przypisane do nich tranzycje. W różnych stanach warunki przejścia mogą się powtarzać, co pozwala na uniknięcie duplikacji kodu. Przykładem może być sprawdzenie, czy użytkownik podał odpowiedź negatywną. Taka funkcja może być wykorzystana podczas obsługi wszystkich pytań zamkniętych oczekujących potwierdzenia lub zaprzeczenia. Dodatkowo kontroler utrzymuje listę, na której umieszcza sprawdzone warunki. Lista pozwala na optymalizację działania maszyny, ponieważ dla każdego warunku sprawdzenie odbywa się najwyżej jeden raz. Nowy stan może na kontrolerze wymusić wyczyszczenie historii, co spowoduje zablokowanie powrotu do poprzednich wypowiedzi.

W celu umożliwienia łatwej zmiany tematu rozmowy zaimplementowane zostało także definiowanie tranzycji globalnych. Nie trzeba wtedy dodawać tej samej funkcji przejścia dla każdego ze stanów zdefiniowanych w maszynie. Tranzycje globalne są sprawdzane na samym końcu, kiedy żaden z warunków stanu bieżącego lub stanów historycznych nie zaakceptował symbolu wejściowego. Przykładem zastosowania tej funkcji może być zdefiniowanie uniwersalnej reakcji na każdy tekst, którego chatbot nie potrafi przetworzyć w inny sposób.

Funkcja przejścia oprócz samego sprawdzania warunków może także zwracać wartość, która jest przekazywana do akcji skojarzonej z nowym stanem. Takie podejście pozwala na optymalizację działania systemu, ponieważ raz wyekstrahowane z wypowiedzi dane mogą być później ponownie użyte.

Opracowana maszyna stanów nie musi reagować jedynie na wypowiedzi użytkownika. Przykładowo chatbot może reagować na milczenie ze strony rozmówcy, próbując zachęcić go do rozmowy lub wyrażać zniecierpliwienie. Takie zachowanie można łatwo zaimplementować, jeśli automat będzie taktowany zegarem o niskiej częstotliwości.

Skuteczność działania powyższego rozwiązania można podnieść poprzez wprowadzenie współczynnika dopasowania. Zamiast przerywać sprawdzanie tranzycji po znalezieniu pierwszej, która akceptuje symbol można tą operację kontynuować przeglądając pozostałe możliwości. Funkcja przejścia w takim podejściu zwracałaby dodatkowo wartość liczbową określającą stopień dopasowania do symbolu wejściowego – przykładowo dla wzorców tekstowych można wykorzystać odległość edycyjną. Kontroler dysponując takimi danymi mógłby wybrać najlepsze dopasowanie, co podwyższyłoby jakość działania całego systemu z punktu widzenia użytkownika.

### 3.5.2. Przetwarzanie wstępne wypowiedzi

Zanim informacja o wypowiedzi użytkownika trafi jako zdarzenie do maszyny stanowej, wprowadzony tekst zostaje poddany jednorazowej obróbce. Jest to kilkustopniowy proces, który pozwoli na bardziej wydajne oraz dokładne działanie mechanizmu dopasowującego. Składa się on z czterech etapów:

- tokenizacja,
- analiza morfologiczna,
- wyszukanie wyrazów oraz fraz bliskoznacznych,
- oznaczenie nazw własnych oraz dat.

Pierwszym etapem przetwarzania tekstu jest tokenizacja. Polega ona na podzieleniu wypowiedzi na pojedyncze słowa – tokeny. Do tego celu zostało wykorzystane wyrażenie regularne, które jednoznacznie definiuje ciąg znaków uznawany za token.

Następną operacją jest przeprowadzenie analizy morfologicznej. Każdemu ze słów zostają przypisane dodatkowe dane, które zostały w parach pozyskane ze słownika:

- forma bazowa – jest to słowo, przez którego odmianę uzyskuje się badany wyraz,
- etykieta fleksyjna – określa część mowy oraz parametry odmiany wyrazu: rodzaj, przypadek, liczbę, osobę, aspekt itp.

Analiza odbywa się na poziomie pojedynczych słów, dlatego jeden wyraz może mieć wiele form bazowych oraz etykiet fleksyjnych. Przykładem takiej niejednoznaczności może być słowo „nie”, które w dwóch poniższych przykładach pochodzi od różnych wyrazów:

- *Nie* widziałem dzisiaj Piotra.
- Małe dzieci są bardzo delikatnie, trzeba szczególnie o *nie* dbać.

Wieloznaczność można wyeliminować wykorzystując otoczenie danego wyrazu. Korzyści wynikające z zaimplementowania analizy morfologicznej z wykorzystaniem kontekstu są jednak dość niewielkie w porównaniu do kosztami, które należy ponieść. Jednym z najważniejszych problemów jest pozyskanie materiału językowego (oznaczonego korpusu), który można byłoby wykorzystać do przeprowadzenia takiej operacji. W związku z tym taka funkcjonalność nie została zrealizowana.

Kolejnym etapem jest odnalezienie synonimów dla wyrazów oraz fraz znajdujących się w tekście. Algorytm realizujący to zadanie pracuje na formach bazowych wprowadzonych słów. Dla każdej z nich jest sprawdzane, czy w słowniku istnieje takie słowo lub fraza, która się nim zaczyna. W przypadku dopasowania całego słowa, jego grupy znaczeniowe są dodawane do informacji na jego temat. Jeśli zostały znalezione frazy, to odbywa się sprawdzenie, czy kolejne słowa także pasują. To dopasowywanie fraz można dodatkowo ulepszyć zastępując próbę sztywnego dopasowania przeglądaniem otoczenia wyrazów, umożliwiając ignorowanie zmiany szyku lub wtrąceń dodatkowych słów.

Podobnie jak w poprzednim kroku, w tym także jest obecny problem niejednoznaczności – dane słowo może należeć do kilku grup znaczeniowych. Przykładem takiego wyrazu może być „zamek”, które można zakwalifikować do dwóch grup:

- budowla – cytadela, forteca, grodzisko, gród, kasztel, twierdza, warownia, zamek,

- mechanizm zabezpieczający – zamek, zamknięcie, zasuw, zatrask.

Ostatnim etapem jest identyfikacja w tekście dat oraz nazw własnych. Wyodrębnianie dat odbywa się z użyciem zestawu wyrażeń regularnych, które starają się wykryć daty określone w sposób absolutny. Przykładem takich dat są: 20 stycznia 2009, rok 1998, 15 maja. Daty względne (2 lata temu, wczoraj) lub określone w bardziej opisowy sposób (rok zakończenia drugiej wojny światowej) nie są rozpoznawane.

Z punktu widzenia transformacji zapytań dla wyszukiwarki bardzo istotną czynnością jest identyfikacja nazw własnych w analizowanym tekście. Do tego celu wykorzystany został lekko zmodyfikowany mechanizm oznaczania synonimów. Za źródło nazw posłużyła lista wszystkich stron Wikipedii, które znajdują się w głównej przestrzeni nazw<sup>3</sup>. Spis stron jest publicznie dostępny w formie skompresowanego pliku tekstowego, którego najnowszą wersję można uzyskać pod adresem <http://download.wikimedia.org/plwiki/latest/plwiki-latest-all-titles-in-ns0.gz><sup>4</sup>. Dodatkowo za nazwę własną jest także uznawane słowo, które nie zostało znalezione w słowniku.

### 3.5.3. Dopasowywanie wzorców

Działanie opracowanego chatbota opiera się na mechanizmie dopasowywania wzorców do tekstu wprowadzonego przez człowieka. Wirtualny rozmówca dysponuje przygotowanym przez człowieka zbiorem reguł, które określają jego zachowanie na podstawie dostarczonych danych wejściowych. Główną zaletą takiego rozwiązania jest jego prostota. Niestety to podejście ma też swoje wady – język naturalny umożliwia wyrażanie jednej myśli na wiele różnych sposobów, dlatego stworzenie uniwersalnego wzorca, który pasowałby do wszystkich nie jest możliwe.

Opracowane zostały trzy różne metody dopasowywania wzorców:

- ścisła – wykorzystanie wyrażeń regularnych dostępnych w Perlu,
- prosta – dopasowywanie kolejnych wyrazów ze wzorca,
- luźna – dopasowywanie słów ze wzorca w luźnym szyku.

Każda z nich jest realizowana przez osobną klasę, która implementuje odpowiedni interfejs. Jest on jednakowy dla wszystkich wzorców, co ułatwia dodawanie nowych algorytmów bez potrzeby modyfikowania reszty systemu. Konstruktor obiektu wzorca ma dostęp do narzędzi przetwarzania języka, więc może on od razu przygotować informacje potrzebne podczas przeszukiwania.

Pierwsza metoda umożliwia przeprowadzenie ścisłego dopasowania – wyrażenia regularnie pozwalają na precyzyjne określenie jaki tekst może zostać dopasowany. Jako wejście może zostać użyty oryginalny tekst podany przez rozmówcę lub też teksty powstałe przez kombinację wszystkich form bazowych i synonimów. Ta metoda nadaje się tylko dla analizowania krótkich wypowiedzi. Luźny szyk zdania w języku polskim znacząco komplikuje tworzenie wyrażeń, które dopasowywałyby więcej niż jeden, konkretny model zdania.

Druga metoda pozwala na podanie listy słów, które mają zostać dopasowane do form podstawowych. Jest to uproszczona wersja metody ścisłej<sup>5</sup>. Algorytm porównuje kolejne słowa ze wzorcem. Jeżeli wszystkie wyrazy znajdują się w tekście, to dopasowanie się powiodło. Algorytm ignoruje nadmiarowe słowa, zostają one dołączone do wyniku dopasowania jako wymagające dodatkowej analizy

<sup>3</sup>Znajdują się w niej wyłącznie hasła, jest to przestrzeń nieposiadająca prefiksu. Więcej informacji na ten temat znajduje się na stronie [http://pl.wikipedia.org/wiki/Pomoc:Przestrzeń\\_nazw](http://pl.wikipedia.org/wiki/Pomoc:Przestrzeń_nazw).

<sup>4</sup>Więcej informacji o zasobach udostępnianych przez fundację Wikimedia zostało zamieszczone w rozdziale 3.6.

<sup>5</sup>Stąd nazwa „prosta”.

przez akcję. Pozwala także na zdefiniowanie czarnej listy słów, które nie mogą zostać zignorowane albo białej listy określającej wszystkie słowa, które będą ignorowane<sup>6</sup>.

Ostatnia metoda jest najbardziej liberalna – pozwala na dopasowywanie słów w dowolnym szyku. Jest ona rozszerzeniem poprzedniej metody o dodatkowy element – odległość edycyjną. Pojęcie to definiuje się jako liczba operacji potrzebnych do transformacji jednego łańcuchu znaków w drugi[11]. Jednym z parametrów wzorca jest jego maksymalna dopuszczalna odległość edycyjna od badanego tekstu. Na początku obliczanie tej wielkości było realizowane algorytmem Wagnera-Fishera[6]. Jednak po kilku próbach okazało się, że dużo lepsze wyniki są osiągnięte jeśli za transformacje są uznawane operacje na całych słowach, a nie pojedynczych znakach. Algorytm można było dodatkowo ulepszyć zastępując trzy domyślne operacje: wstawienia, usunięcia, zamiany. Zakładając, że transformacji poddawany będzie tekst wprowadzony przez użytkownika zostały wprowadzone nowe operacje: usunięcie nadmiarowego słowa, inne odmienienie wyrazu oraz przesunięcie słowa lub frazy w inne miejsce. Przy tak określonych warunkach może się zdarzyć, że przeprowadzenie transformacji nie jest wykonalne, wtedy dopasowanie zostaje uznane za nieudane. Działanie tego mechanizmu można dodatkowo dostrajać nadając różne koszty każdej z operacji.

Innym zadaniem mechanizmu dopasowywania wzorców jest ekstrakcja informacji. Przykładowo w perlowych wyrażeniach regularnych używając odpowiednich metasymboli można wskazywać, które fragmenty tekstu powinny zostać „wycięte”<sup>7</sup>. Pozyskane w ten sposób dane są przekazywane dalej, do metody obsługującej daną akcję.

Powyższe rozwiązania można dodatkowo usprawnić wprowadzając bardziej rozbudowany, tekstowy język opisu wzorców, który czerpałby z wyrażeń regularnych. Różniłby się od nich tym, że mechanizm dopasowania operowałby na słowach uwzględniając aspekty związane z językiem naturalnym. Przykładowy wzorec „([:adj:]+) [:subst:]@kot“ mógłby ekstrahować jeden lub więcej słów będących przymiotnikami, za którym występuje dowolnie odmieniony rzeczownik „kot”<sup>8</sup>. Dodatkową funkcjonalnością byłoby grupowanie wyrazów, które powinny występować obok siebie, natomiast same grupy mogłyby występować w dowolnej kolejności.

### 3.5.4. Odpowiadanie

Chatbot oprócz analizowania wypowiedzi użytkownika powinien także na nie odpowiadać. Do tego celu wykorzystuje zbiór szablonów, które zostały przygotowane przez człowieka. Każda akcja posiada przypisanych co najmniej kilka różnych wzorów. Przed wysłaniem odpowiedzi szablon jest dobierany w sposób losowy, a następnie wypełniany danymi. Takie działanie sprawia, że rozmowa nie wygląda zawsze w ten sam sposób.

Szablony powinny być jak najbardziej uniwersalne, dlatego czasem zachodzi potrzeba odmienienia wstawianych wyrazów, tak aby pasowały do kontekstu. Algorytm wypełniający szablony posługuje się w tym celu opracowanym słownikiem, który umożliwia odmianę słów według zadanych parametrów (etykiety fleksyjnej).

Wirtualny rozmówca wykorzystujący w odpowiedziach informacje podane wcześniej przez użytkownika sprawia wrażenie bardziej inteligentnego. W związku z tym szablony mogą zawierać odwołania do pól znajdujących się w specjalnej tablicy asocjacyjnej, która opisuje rozmówcę. Jest ona stopniowo wy-

<sup>6</sup>Użycie jednej listy wyklucza korzystanie z drugiej.

<sup>7</sup>Dokładne informacje na temat działania mechanizmu wyrażeń regularnych są zamieszczone na stronie perlre podręcznika perlloc.

<sup>8</sup>Zaprezentowana notacja jest tylko sugestią.

pełniana danymi podczas prowadzenia rozmowy. Jeśli użytkownik przedstawił się (Cześć, jestem Jarek), to chatbot może go pożegnać używając imienia rozmówcy (Do zobaczenia Jarku!).

Podczas rozmów z użytkownikami na pewno będzie się zdarzać sytuacja, w której nie uda się dopasować żadnego ze wzorców. Wynika to z tego, że nie jest możliwe opisanie regułami wszystkich wariantów wypowiedzi. Potrzebny jest algorytm, który powinien obsługiwać takie przypadki. Opracowany chatbot podejmuje próbę odgadnięcia intencji użytkownika. Odbywa się to poprzez sprawdzenie jak odmienione zostały czasowniki w wypowiedzi. Jeżeli tekst zawiera czasownik w trzeciej osobie lub go w ogóle nie zawiera to jest traktowany jako zapytanie. Na tej podstawie dokonywana jest aktualizacja kryteriów wyszukiwania<sup>9</sup>. Natomiast kiedy tekst zawiera czasowniki w drugiej osobie oraz ma formę pytania to chatbot będzie starał się żartobliwie odpowiedzieć na to pytanie, przykład:

– Czy lubisz obwarzanki?

– Niestety nie mogę udzielić odpowiedzi na to pytanie, jest to informacja niejawna :-).

W pozostałych przypadkach, kiedy wszystkie metody zawiodły, użytkownik zostanie poinformowany o tym, że nie został zrozumiany i poproszony o wyrażenie się w inny sposób.

### 3.6. Wyszukiwarka

Głównym zadaniem modułu wyszukiwarki jest koordynacja całego procesu pozyskiwania oraz klasyfikacji dokumentów. Podczas projektowania mechanizmu wyszukującego zostały wzięte pod uwagę trzy różne sposoby jego implementacji.

- Opracowanie od podstaw pająka oraz indeksera, a następnie zbudowanie indeksu stron.
- Przygotowanie samego pająka oraz wykorzystanie ogólnodostępnego oprogramowania do indeksowania dokumentów.
- Transformacja opisu z języka naturalnego na język zapytań, wykorzystanie ogólnodostępnych serwisów wyszukiwających, a następnie przetworzenie i odfiltrowanie uzyskanych wyników.

W celu zbudowania indeksu pająk musi pobrać wszystkie strony z artykułami, co wymaga łącza internetowego o dobrej przepustowości oraz odpowiedniej ilości miejsca na dysku. Ze względu na szczególny charakter Wikipedii zbudowanie indeksu od zera może zostać nieco ułatwione, ponieważ fundacja Wikimedia udostępnia treść nie tylko za pośrednictwem tradycyjnych stron WWW.

Pod adresem <http://static.wikipedia.org> dostępne są skompresowane archiwa ze statycznym zbiorem stron HTML dla każdego języka. Wykorzystanie takiego archiwum pozwoliłoby w dużym stopniu zaoszczędzić czas oraz łącze internetowe, ponieważ nie trzeba wtedy pobierać stron pojedynczo. Niestety nowe archiwa nie są już tworzone, a ostatnie były wykonane w czerwcu 2008 roku. Indeks zbudowany na bazie takiego zbioru musiałby później i tak zostać w całości zaktualizowany, co wykluczyło tę możliwość.

Drugim potencjalnym źródłem danych mogły być zrzuty bazy danych udostępniane w formacie XML pod adresem <http://download.wikimedia.org>. Są one tworzone regularnie i w różnych wersjach – można sobie wybrać plik, który najbardziej odpowiada potrzebom. Do niniejszej pracy najbardziej odpowiednie jest archiwum `pages-articles.xml.bz2` zawierające ostatnie wersje artykułów, szablonów, opisów grafik oraz głównych meta-stron. Niestety to źródło także nie jest całkowicie pozbawione wad – kod źródłowy każdej strony jest zapisany przy użyciu składni wiki. Aby uzyskać

<sup>9</sup>Więcej informacji na ten temat transformacji zapytań znajduje się w rozdziale 3.6.1.

tekst, który oglądają zwykli czytelnicy ten zrzut należałoby zaimportować do bazy MySQL, a następnie wykorzystując skrypt `dumpHTML.php` (dostarczany z oprogramowaniem MediaWiki) wyeksportować do kodu HTML.

Alternatywą dla budowania indeksu jest wariant bazujący na wykorzystaniu istniejących serwisów wyszukiwujących. Przykładowy sposób działania takiego rozwiązania może być następujący:

- Transformacja uzyskanych informacji od użytkownika na język zapytań.
- Wysłanie zapytań do wyszukiwarek.
- Zebranie uzyskanych wyników.
- Pobranie stron oraz ekstrakcja tekstu.
- Przeprowadzenie klasyfikacji oraz oceny trafności przydatności każdego z dokumentów.

Należy mieć na uwadze pewne ograniczenia, które wiążą się z angażowaniem serwisów zewnętrznych. Takiego rozwiązania nie można stosować na treściach, które nie są publicznie dostępne (przeszukiwanie dysku lokalnego komputera, dokumentów firmowych). Czas odpowiedzi na zapytanie wydłuża się, ponieważ strony trzeba pobrać, a następnie przetworzyć. Takie opóźnienia można częściowo zredukować stosując agresywne buforowanie. Serwisy mogą też posiadać limit na liczbę zapytań z jednego adresu IP, po którego przekroczeniu nastąpi odmowa dalszej obsługi<sup>10</sup>.

Ze względu na ograniczone zasoby, do zaimplementowania wybrany został ostatni wariant – wykorzystanie istniejących serwisów. Na potrzeby pracy takie podejście w zupełności wystarcza, a jego główną zaletą jest fakt, że nie trzeba tworzyć, przechowywać oraz utrzymywać indeksu o dość dużych rozmiarach. Następujące serwisy wyszukiwujące były brane pod uwagę podczas wyboru:

- Google – najpopularniejszy dostawca usług wyszukiwania; oficjalne API umożliwiające pobieranie wyników zapytań oparte o protokół SOAP zostało wycofane w grudniu 2006 roku, od tamtej pory wyszukiwarki nie można wykorzystywać w sposób zautomatyzowany<sup>11</sup>.
- Bing – wyszukiwarka opracowana przez firmę Microsoft, poprzednio znana jako: Windows Live Search oraz MSN Search; oferuje programistom API, jednak z dużymi ograniczeniami<sup>12</sup>.
- Yahoo! Search – wyszukiwarka ogólnego przeznaczenia, od 2009 roku wykorzystuje infrastrukturę wyszukiwarki Bing; udostępnia usługę BOSS (ang. *Build your Own Search Service*), pozwalającą na budowanie własnych rozwiązań wyszukiwawczych.
- Wikiwix – wyspecjalizowana wyszukiwarka artykułów w Wikipedii, jej polskojęzyczna wersja jest dostępna pod adresem <http://pl.wikiwix.org>; udostępnia API dla posiadaczy specjalnego klucza,
- Wyszukiwarka oferowana przez Wikipedię – wykorzystująca jako silnik oprogramowanie Apache Lucene, dostępna poprzez api pod adresem <http://pl.wikipedia.org/w/api.php>.

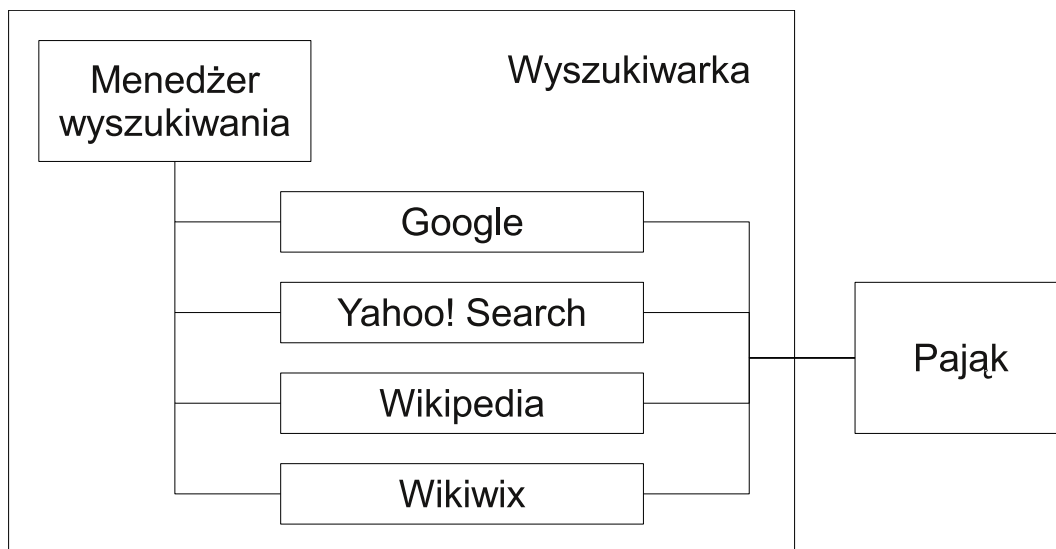
<sup>10</sup>Przykładem takiej wyszukiwarki może być Google. Po stwierdzeniu dużej liczby zapytań wysłanych w bardzo krótkim czasie użytkownik zamiast wyników otrzymuje komunikat informujący go o blokadzie. Zablokowany internauta chcący otrzymać wyniki musi udowodnić, że jest się człowiekiem przepisując tekst z obrazka.

<sup>11</sup><http://www.google.pl/accounts/TOS> – Warunki korzystania z Serwisów spółki Google, punkt 5.3: „Użytkownik zgadza się nie uzyskiwać (ani próbować uzyskać) dostępu do Serwisów przy pomocy innych środków niż interfejs udostępniony przez spółkę Google, chyba że Użytkownik uzyskał od spółki Google w drodze odrębnej umowy zgodę na taki dostęp.”

<sup>12</sup><http://explore.live.com/microsoft-service-agreement?ref=none> – Microsoft Service Agreement, punkt 4, „You must not: (...) or use any automated process or service (such as a bot, a spider, periodic caching of information stored by Microsoft, or metasearching) to access or use the service.”



Do implementacji zostały wybrane cztery wyszukiwarki: Wikiwix, wyszukiwarka Wikipedii, Yahoo! Search oraz Google. Modułarna architektura całej aplikacji pozwala łatwo zmieniać i dodawać wykorzystywanych dostawców usług wyszukiwania, ponieważ każdy moduł implementuje taki sam interfejs. Co więcej, aplikacja może używać kilku dostawców jednocześnie. Wyjątek stanowi wyszukiwarka Google, której obsługa została zaimplementowana wyłącznie w celach porównawczych, na potrzeby tej pracy – wynika to z ograniczeń, o których wspomniano wcześniej.



Rysunek 3.4: Relacje między wybranymi komponentami.

Wszystkie operacje wyszukiwania są wykonywane za pośrednictwem obiektu pełniącego rolę Menedżera wyszukiwań. Zajmuje się on wysyłaniem zapytań do modułów poszczególnych dostawców, a następnie zbieraniem wyników. Każdy z dostawców wysyła żądania HTTP do pająka, który je realizuje. Pająk oprócz serwera WWW jest jedynym modułem, który komunikuje się ze światem zewnętrznym. To on zapewnia buforowanie stron w pamięci podręcznej, ograniczanie liczby żądań na dany adres oraz kolejkovanie zadań. Tworzy to swego rodzaju strukturę warstwową obiektów – dany komponent dla obiektu z niższej warstwy jest konsumentem informacji, natomiast dla obiektu warstwy wyższej producentem. Pomiedzy nimi asynchronicznie przepływają komunikaty.

### 3.6.1. Transformacja zapytań

Użytkownik korzystający z systemu opisuje swoją potrzebę informacyjną w języku naturalnym. Zadaniem systemu jest określenie kryteriów wyszukiwania na podstawie zebranych danych. Są one wykorzystywane do zlokalizowania zasobów, które zawierają pożądaną przez rozmówcę treści.

Nie wszystkie elementy wypowiedzi człowieka niosą informacje o obiekcie poszukiwań. Formy grzecznościowe to jeden z przykładów – nie zawierają informacji, jednak są częścią większości konwersacji. Przed przystąpieniem do transformacji zapytań takie zbędne elementy należy odfiltrować. W przeciwnym razie nadmiarowe wyrażenia mogą zakłócać przebieg wyszukiwania. Poniżej zostały zamieszczone przykłady zdań, gdzie niepotrzebne fragmenty tekstu zostały wyróżnione:

- *Witaj, czy możesz mi podać jak się nazywa prezydent USA?*
- *Czy wiesz, w którym województwie znajduje się Kraków?*

Czasem może się zdarzyć, że zignorowane zostaną całe wypowiedzi. Do filtracji tekstu wykorzystywane są odpowiednio przygotowane wzorce. Sposób ich działania został opisany w rozdziale 3.5.3.

Pierwszym etapem transformacji jest próba określenia, czego dotyczy poszukiwana informacja. Mechanizm ten sprawdza, czy wypowiedzi użytkownika zawierają elementy, na podstawie których da się to jednoznacznie stwierdzić. Poniżej została zaprezentowana tabelka z przykładowymi słowami:

Przykładowe słowa	Rodzaj poszukiwanej informacji
gdzie, w jakim mieście, wskaż lokalizację	miejsce
kiedy, podaj datę, w którym roku	data
kto, jak ma na nazwisko	osoba

Tablica 3.1: Przykładowe słowa określające rodzaj poszukiwanej informacji.

Kolejnym etapem jest umiejscowienie poszukiwanej informacji w czasie – czy dotyczy ona przeszłości, teraźniejszości, czy przyszłości. Odbywa się to poprzez sprawdzenie w słowniku, jak zostały odmienione czasowniki w wypowiedziach użytkownika. Poniżej znajduje się tabelka z przykładem, jak zmieniają się oczekiwania w zależności od czasu.

Pytanie	Oczekiwany rezultat
Kto <i>był</i> prezydentem Polski?	lista prezydentów Polski
Kto <i>jest</i> prezydentem Polski?	informacje na temat obecnego prezydenta
Kto <i>będzie</i> prezydentem Polski?	oficjalna lista kandydatów na prezydenta

Tablica 3.2: Wpływ odmiany czasownika na oczekiwane rezultaty wyszukiwania.

Następnym etapem jest stworzenie listy słów oraz fraz, które powinny się znajdować w dokumencie z poszukiwaną informacją. Jest ona wykorzystywana podczas generowania zapytań do zewnętrznych wyszukiwarek oraz klasyfikacji dokumentów. Słowom oraz frazom są przyporządkowywane wagi, na podstawie których są one wybierane do budowania zapytań. Poniżej jest przedstawiona lista klas, uporządkowana malejąco według stopnia ważności:

- frazy składające się co najmniej z rzeczownika oraz czasownika,
- nazwy własne,
- daty,
- synonimy we frazach składających się co najmniej z rzeczownika oraz czasownika,
- synonimy nazw własnych
- rzeczowniki,
- synonimy rzeczowników,
- czasowniki,
- synonimy czasowników,
- pozostałe wyrazy.

Z tej listy wykluczone są słowa pospolite (ang. *stop-words*), które mogą zaburzać wyniki wyszukiwania. Umieszczenie fraz składających się co najmniej z rzeczownika oraz czasownika na pierwszym miejscu

wynika z tego, że udane dopasowanie ciągu słów będzie rzadsze, a to oznacza wyższe prawdopodobieństwo trafienia na poszukiwaną informację. Nazwy własne otrzymały drugie miejsce, ponieważ użytkownik może napisać w sposób bezpośredni o czym potrzebuje informacje. Przykładem takich pytań są:

- Gdzie urodził się Jan Matejko?
- Gdzie znajduje się Poznań?

Przekazanie danych do menedżera wyszukiwania odbywa się dopiero, kiedy lista zawiera dane pozwalające na rozpoczęcie poszukiwań. Do takich danych zaliczane są nazwy własne, frazy oraz daty. Zapytania generowane są kolejno według listy przedstawionej powyżej. Lista jest ciągle uzupełniana o nowe elementy, w miarę napływania nowych informacji od użytkownika. Może się zdarzyć, że po zapytaniu składającym się z daty pojawi się nazwa własna, która jeszcze nie była sprawdzana, a która niedawno padła podczas rozmowy.

### 3.6.2. Klasyfikacja dokumentów

Zanim pobrane dokumenty zostaną poddane jakiegokolwiek analizie muszą zostać wcześniej odpowiednio przetworzone. Proces ten składa się z kilku czynności:

- Określenie standardu w jakim dokument został zakodowany. Większość dokumentów posiada taką informację. Jest ona umieszczona w polu *Content-type* nagłówka odpowiedzi protokołu HTTP. Jego odpowiednik może znajdować się także w kodzie dokumentu HTML jako znacznik *meta* sekcji nagłówkowej z parametrem *http-equiv*. Niekiedy może się zdarzyć, że autor strony nie zdefiniował kodowania – przeglądarki internetowe używają pewnych heurystyk w celu określenia zestawu znaków, jednak w tym rozwiązaniu takie strony będą ignorowane.
- Usunięcie kodu HTML z dokumentu wraz z informacjami, które nie są istotne z punktu widzenia procesu klasyfikacji – pola nagłówka, skrypty w języku JavaScript, komentarze, style CSS.
- Konwersja tekstu do kodowania utf-8, jeśli jest ono inne.
- Zmiana encji na znaki przez nie reprezentowane (np. &copy; na ©).

Otrzymany tekst jest następnie poddawany analizie, która pozwoli na sprawne przeprowadzenie klasyfikacji. Można ją podzielić na kilka odrębnych zadań:

- tokenizacja,
- analiza morfologiczna,
- wyszukanie wyrazów oraz fraz bliskoznacznych,
- oznaczenie dat.

Opis powyższych czynności został zawarty w sekcji 3.5.2, która dotyczy wstępnego przetwarzania wypowiedzi użytkownika.

Do oceny dokumentów niezbędna jest metryka, która pozwoli na porównywanie dokumentów między sobą. W związku z tym, na potrzeby tej pracy została opracowana metoda obliczania wartości liczbowej, która określa stopień zaspokojenia potrzeby informacyjnej dla zadanych kryteriów.

Klasa	Waga
fraza składająca się co najmniej z rzeczownika oraz czasownika	20
nazwa własna	19
data	18
synonim frazy składającej się co najmniej z rzeczownika oraz czasownika	15
synonim nazwy własnej	14
rzeczownik	8
synonim rzeczownika	7
czasownik	5
synonim czasownika	4
inny wyraz	1

Tablica 3.3: Przykładowe przyporządkowanie wag poszczególnym klasom słów.

Algorytmu obliczający ocenę polega głównie na sprawdzaniu, czy w badanym tekście występują słowa, które są uznawane za kryteria wyszukiwania. Przeszukiwanie rozpoczyna się od elementów, które mają najwyższą wagę. W tabeli 3.6.2 zostały przedstawione wagi dla poszczególnych klas. Kiedy uda się dopasować dane słowo lub frazę, to do oceny dokumentu jest dodawana waga przyporządkowana dla klasy, do której szukany tekst należy. Wszystkie elementy z niższych klas, które są jego pochodnymi nie są już brane pod uwagę. Przykładowo, jeżeli w tekście została odnaleziona nazwa własna „Warszawa“, to algorytm nie będzie już sprawdzał obecności synonimu tej nazwy własnej jakim jest „stolica Polski” i innych. Analogicznie jeśli w dokumencie została dopasowana fraza „urodził się w Gdańsku“, słowo Gdańsk nie będzie już sprawdzane.

Powyższą metodę można rozszerzyć o sprawdzanie w jakiej odległości od siebie leżą poszczególne fragmenty tekstu i na tej podstawie modyfikować wynik obliczeń. Pozwoliłoby to na ekstrakcję fragmentu dokumentu, który może zawierać poszukiwaną informację. Chatbot odpowiadający na pytania mógłby próbować skonstruować odpowiedź parafrazując znaleziony tekst.

Po sprawdzeniu wszystkich kryteriów dokumentowi zostaje przypisana obliczona suma. Cała operacja jest powtarzana dla każdej z pobranych stron. Kiedy wszystkie dokumenty zostały pobrane i ocenione następuje przejście do kolejnego etapu klasyfikacji – wybór najlepszych wyników. Lista artykułów zostaje uporządkowana malejąco według obliczonej wartości. Sam fakt, że dokument znajduje się na pierwszym miejscu nie oznacza, że zawiera on poszukiwane informacje. Potrzebna jest zatem możliwość stwierdzenia w jakim stopniu znaleziony dokument odpowiada potrzebom użytkownika. W tym celu została wprowadzona normalizacja ocen, tak aby znajdowały się one w przedziale  $[0, 1]$ . Odbywa się to przez określenie maksymalnej wartości, jaką mógłby otrzymać dokument, a następnie podzielenie każdej oceny przez to maksimum.

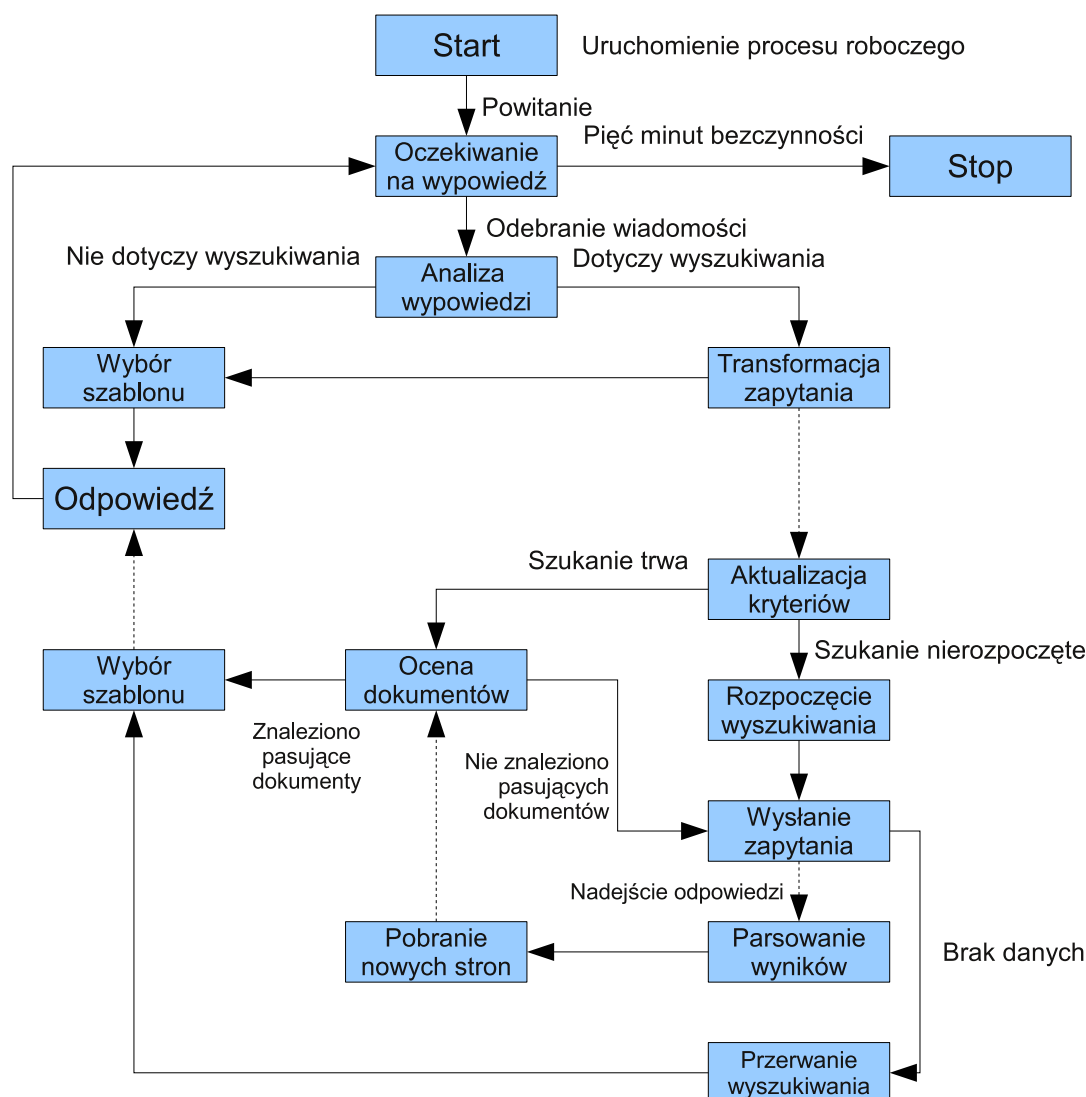
Ograniczenie ocen do liczb z przedziału  $[0, 1]$  pozwala na ustalenie procentowych progów, po których osiągnięciu wyniki są prezentowane użytkownikowi. Każdemu przedziałowi jest przypisana odrębna akcja:

- powyżej 80% – prezentacja adresu znalezionej strony; jeżeli stron jest więcej, wyświetlenie adresów trzech najwyższej ocenionych stron.
- powyżej 50% – prezentacja adresów znalezionych stron, wraz z uwagą, że mogą one nie zawierać poszukiwanych informacji.
- pozostałe – komunikat, że nie znaleziono informacji na wybrany temat.

Czasem będzie zachodzić sytuacja, w której do pierwszej kategorii zostało zakwalifikowana bardzo duża liczba stron. Kiedy takich dokumentów jest więcej niż dziesięć, chatbot informuje rozmówcę, że powinien dokładniej opisać poszukiwaną informację. Stanowi to ważny element procesu wyszukiwania, ponieważ dla prostych zapytań typu „Co to jest kot?”, czy „Kim był Jan?” można znaleźć setki stron. Wyświetlanie tak długiej listy przez chatbota jest niepraktyczne.

### 3.7. Praca aplikacji

Wcześniejsze rozdziały traktowały ogólnie o budowie aplikacji oraz szczegółowo o najważniejszych komponentach. W tym rozdziale znajduje się krótki opis przedstawiający pracę systemu jako całości.



Rysunek 3.5: Uproszczony schemat działania aplikacji.

Rysunek 3.5 przedstawia w uproszczony sposób mechanizm działania aplikacji. Po wejściu użytkownika na stronę wyszukiwarki do jego obsługi zostają przydzielone zasoby. Chatbot wysyła komunikat powitalny, a następnie czeka na reakcję ze strony swojego rozmówcy. Użytkownik wysyłając tekst wywołuje swoim działaniem akcję w systemie. Kiedy przez określony czas żadna akcja nie zostanie wywołana, proces obsługujący użytkownika zostaje zakończony.

Wypowiedzi można podzielić na dwie kategorie:

- niezwiązane z wyszukiwaniem,
- określające potrzebę informacyjną.

Kiedy rozmowa nie dotyczy głównego zadania chatbota jakim jest wyszukiwanie informacji, jego reakcje sprowadzają się głównie do dopasowania wzorca, wyboru szablonu z odpowiedzią, wypełnienia go oraz odesłania użytkownikowi. Wyekstrahowane dane z otrzymanych wiadomości zostają dodatkowo zapisane w tablicy asocjacyjnej, która jest wykorzystywana podczas konwersacji.

Właściwa praca systemu rozpoczyna się, kiedy rozmówca zaczyna określać jakich informacji potrzebuje. Reakcją chatbota na pierwszą tego typu wiadomość jest odpowiedź, że wyszukiwanie się rozpoczęło. Towarzyszy jej zachęta do rozwinięcia opisu pożądanego wiedzy. W tle natomiast zaczyna się przetwarzanie otrzymanych wymagań. Opis użytkownika zostaje przetransformowany na zapytania, które zostają wysłane do zewnętrznych dostawców usług wyszukiwania. Otrzymane wyniki są filtrowane, a następnie grupowane. Lista stron zostaje przekazana do pająka, którego zadaniem jest ich pobranie. Każdy ściągnięty dokument zostaje poddany klasyfikacji. Rezultatem tego procesu jest lista artykułów uporządkowana według stopnia zgodności z wymaganiami. Jeżeli istnieją pojedyncze dokumenty, które mają odpowiednią ocenę to ich lista jest prezentowana jako wynik użytkownikowi. W przypadku, kiedy takich stron jest dużo, chatbot informuje, że wymagania nie są precyzyjne i prosi o podanie szczegółów, które pozwoliłyby na zawężenie obszaru poszukiwań.

Rozmówca podając nowe wymagania powoduje aktualizację istniejących kryteriów oraz ponowną klasyfikację dokumentów. Kiedy wyniki w dalszym ciągu nie są zadowalające nowe zapytania są kierowane do zewnętrznych wyszukiwarek i cały proces się powtarza.

## 4. Prezentacja oraz ocena działania aplikacji

W tej części pracy znajduje się opis aplikacji widzianej oczami użytkownika. W rozdziale 4.1 został opisany graficzny interfejs. Następnie rozdział 4.2 prezentuje działanie aplikacji wykorzystując przykładowe rozmowy z chatbotem. Rozdział 4.3 przedstawia ocenę funkcjonowania stworzonego oprogramowania w porównaniu z innymi tego typu rozwiązaniami.

### 4.1. Interfejs użytkownika

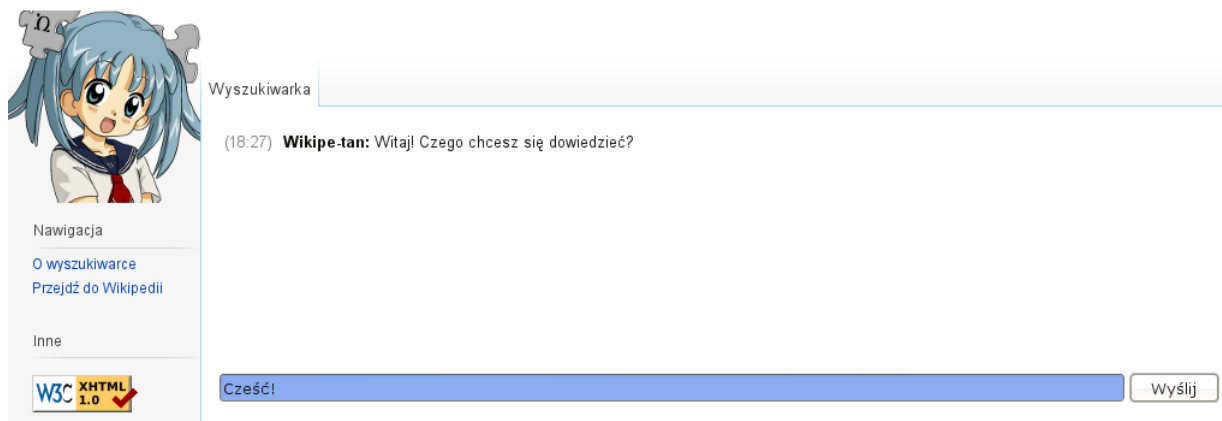
Część kliencka aplikacji składa się z jednej, bardzo prostej strony, której wygląd jest wzorowany na skórcie Vector oprogramowania MediaWiki<sup>1</sup>. Rysunek 4.2 ukazuje wygląd serwisu zaraz po jego otworzeniu. Strona została podzielona na dwie części. Po prawej stronie znajduje się obszar rozmowy, w którym są wyświetlane wypowiedzi. Z wyglądu przypomina czat lub komunikator internetowy – każda wypowiedź ma przypisany identyfikator rozmówcy oraz godzinę. Dodatkowo wypowiedzi sąsiadujące ze sobą wyświetlane są na różnych tłach. Na dole strony znajduje się pole do wprowadzania tekstu oraz opcjonalny przycisk *Wyślij*. Po lewej stronie znajduje się pasek nawigacyjny. Pozwala on na bezpośrednie przejście na strony Wikipedii lub wyświetlenie informacji na temat samej aplikacji. W lewym górnym rogu została umieszczona grafika reprezentująca rozmówcę. Zamiast zdjęcia rzeczywistej osoby wybrany został obrazek narysowany w stylu japońskiego komiksu. Przedstawia on Wikipę-tan – nieoficjalną maskotkę Wikipedii stworzoną przez japońskiego wikipedystę o pseudonimie Kasuga<sup>2</sup>.



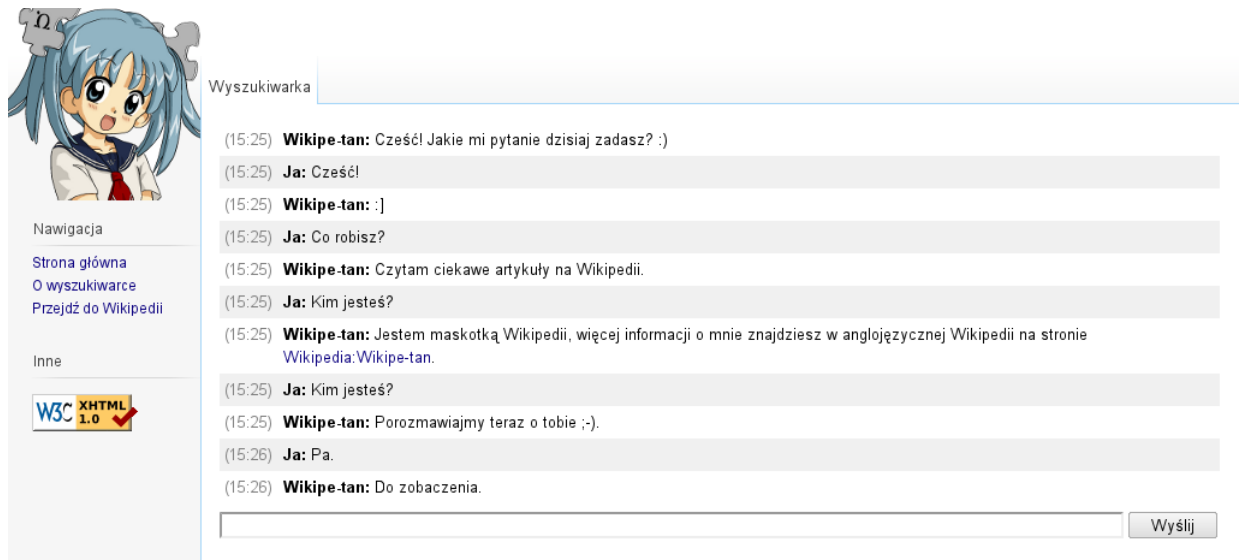
Rysunek 4.1: Wikipę-tan

<sup>1</sup>Oryginalny wygląd można obejrzeć na stronie <http://pl.wikipedia.org/wiki/?useskin=vector>.

<sup>2</sup>Więcej informacji o Wikipę-tan można znaleźć w języku angielskim pod adresem <http://en.wikipedia.org/wiki/Wikipedia:Wikipę-tan>.



Rysunek 4.2: Interfejs wyszukiwarki – rozpoczęcie konwersacji.



Rysunek 4.3: Prowadzenie konwersacji.

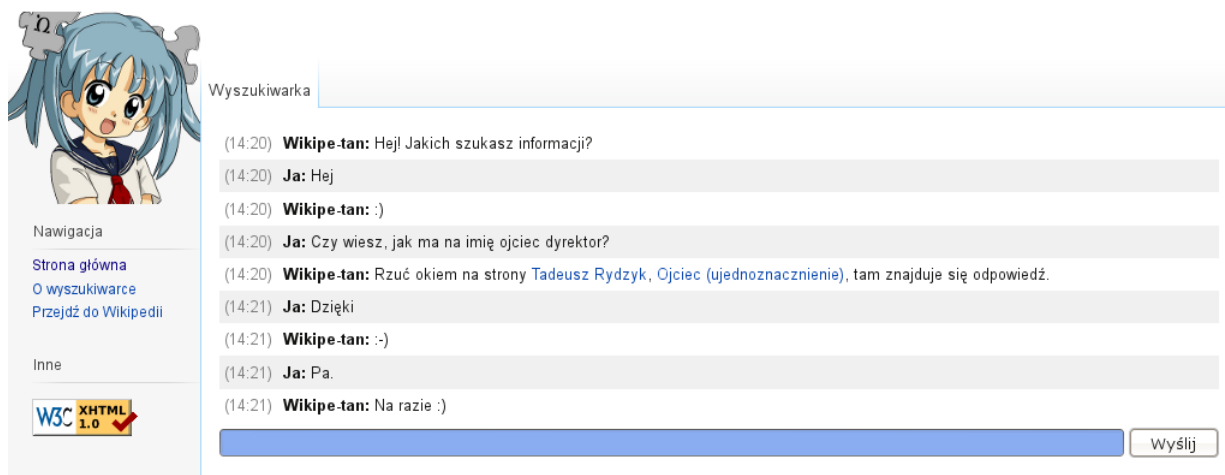
Strona do poprawnego działania wymaga przeglądarki obsługującej skrypty napisane w języku JavaScript. Podczas rozmowy strona nie jest przeładowywana, a zapytania są wysyłane z wykorzystaniem technologii AJAX. Działanie strony było testowane na przeglądarkach: Mozilla Firefox 3.6, Opera 10, Chromium 6, Konqueror 4.5. Przeglądarka Microsoft Internet Explorer nie była sprawdzana, ponieważ nie ma wydania działającego na systemach GNU/Linux. Próba zainstalowania najnowszej wersji przeglądarki z wykorzystaniem środowiska Wine zakończyła się niepowodzeniem.

## 4.2. Wybrane rozmowy

Rozmowy prowadzone z chatbotem będą bardzo krótkie, ponieważ jego głównym zadaniem jest wyszukiwanie informacji, a nie dostarczanie rozrywki. Na rysunku 4.3 został przedstawiony mechanizm prowadzenia dialogu.

- `Wikipie-tan: Cześć! Jakie mi pytanie dzisiaj zadasz? :)`  
Jest to standardowe powitanie użytkownika, na które rozmówca może odpowiedzieć jeśli chce.
- `Ja: Cześć!`  
`Wikipie-tan: :]`  
Jeśli chatbot zostanie przywitany, to zazwyczaj odpowiada emotikoną.
- `Ja: Co robisz?`  
`Wikipie-tan: Czytam ciekawe artykuły na Wikipedii.`  
`Ja: Kim jesteś?`  
`Wikipie-tan: Jestem maskotką Wikipedii, więcej informacji o mnie znajdziesz w anglojęzycznej Wikipedii na stronie Wikipedia:Wikipie-tan.`  
Chatbot ma przygotowane kilka odpowiedzi na pytania dotyczące jego samego.
- `Ja: Kim jesteś?`





Rysunek 4.4: Proste wyszukiwanie.

Wikipie-tan: Porozmawiajmy teraz o tobie ;-).

Zaimplementowana maszyna stanowa umożliwia w prosty sposób sprawdzenie, czy użytkownik się powtarza i reagowanie na to zdarzenie w odpowiedni sposób.

- Ja: Pa.  
Wikipie-tan: Do zobaczenia.
- Chatbot umie się także pożegnać.

Nie jest możliwe przygotowanie odpowiedzi na wszystkie pytania dotyczące chatbota, dlatego musi istnieć mechanizm, który pozwala na obsłużenie takich sytuacji.

- Wikipie-tan: Witaj! Czego chcesz się dowiedzieć? :-)  
Powitania są losowane, tak aby rozmowy nie wyglądały zawsze w ten sam sposób.
- Ja: Jak masz na imię?  
Wikipie-tan: Nie uważasz, że to trochę głupie pytanie?  
Jest to wcześniej przygotowana odpowiedź.
- Ja: Czy lubisz śpiewać?  
Wikipie-tan: Niestety nie mogę udzielić odpowiedzi na to pytanie, jest to informacja niejawna :-).  
Jeśli pytanie dotyczy chatbota, a nie pasuje ono do żadnego z wzorców, to zostaje uruchomiona uniwersalna akcja.

Na rysunku 4.4 zostało przedstawione proste wyszukiwanie. Przykładowa rozmowa:

- Wikipie-tan: Witaj! Czego chcesz się dowiedzieć? :-)  
Ja: Heja.
- Chatbot nie zawsze reaguje na odpowiedź do powitania.

- Ja: Które miejsce zajął Małysz w Salt Lake City?  
Wikip-e-tan: Informacja, której szukasz znajduje się na stronach Skoki narciarskie na Zimowych Igrzyskach Olimpijskich 2002, Zimowe Igrzyska Olimpijskie 2002.

Po zadaniu pytania rozpoczął się proces wyszukiwania. Chatbot prawidłowo wytypował dwie strony, na których znajduje się odpowiedź na postawione pytanie.

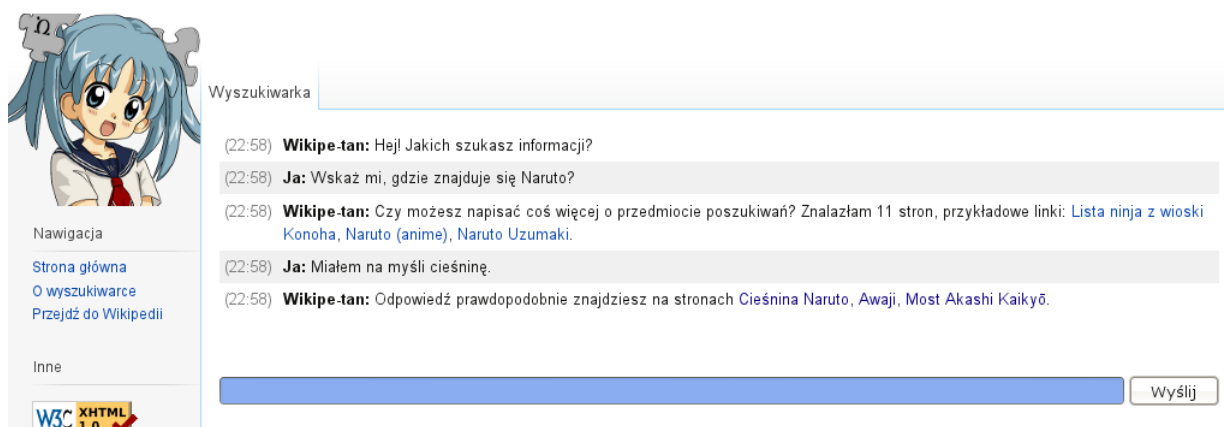
- Ja: Super, dzięki.  
Wikip-e-tan: Zawsze do usług.

Przykładowe pytania:

- Ja: Kto jest autorem Mazurka Dąbrowskiego?  
Wikip-e-tan: Rzuć okiem na strony Józef Wybicki, Komety, Parkowo, tam znajduje się odpowiedź.  
Pierwsza i ostatnia strona zawiera informacje o Józefie Wybickim. Druga strona opisuje polski zespół rockowy, którego jeden z utworów także nosi tytuł *Mazurek Dąbrowskiego*.
- Ja: Czy wiesz, w którym województwie znajduje się Kraków?  
Wikip-e-tan: Informacja, której szukasz znajduje się na stronach Województwo, Województwo małopolskie, Województwo śląskie.  
Pierwsze dwie strony zawierają poszukiwaną informację. Trzecia stanowi pomyłkę, która wynika z tego, że na niej znajdują się informacje o trasach komunikacyjnych, które biegną także przez Kraków.
- Ja: Gdzie leży Gdańsk?  
Wikip-e-tan: Strony Gdańsk, Trójmiasto, Port Gdańsk zawierają informację, której potrzebujesz.  
Czasem chatbot znajduje więcej pasujących stron niż trzy, jednak jeśli liczba ta nie przekracza 10, to są prezentowane pierwsze trzy wyniki.

Zadaniem chatbota jest także obsługiwanie sytuacji, w których podane przez człowieka informacje są niewystarczające do udzielenia jednoznacznej odpowiedzi (rys. 4.5).

- Wikip-e-tan: Hej! Jakich szukasz informacji?  
Ja: Wskaż mi, gdzie znajduje się Naruto?  
Nazwa Naruto jest w tym kontekście niejednoznaczna.
- Wikip-e-tan: Czy możesz napisać coś więcej o przedmiocie poszukiwań?  
Znalazłam 11 stron, przykładowe linki: Lista ninja z wioski Konoha, Naruto (anime), Naruto Uzumaki.  
Chatbot informuje użytkownika, że nie jest w stanie dobrze wytypować stron i prosi ujednoznacznienie zapytania. Zaprezentowane trzy strony dotyczą japońskiego filmu animowanego o tytule Naruto i jego tytułowej postaci.



Rysunek 4.5: Ujednoznacznienie wyszukiwania.

- Ja: Miałem na myśli cieśninę.

Wikipie-tan: Odpowiedź prawdopodobnie znajdziesz na stronach *Cieśnina Naruto*, *Awaji*, *Most Akashi Kaikyou*.

Uzupełnienie informacji powoduje wznowienie wyszukiwania, którego rezultatem są trzy strony zawierające poszukiwaną informację.

Inny przykład zapytania wymagającego ujednoznacznienia:

- Wikipie-tan: Witaj! Czego chcesz się dowiedzieć? :-)

Ja: Gdzie występuje antylopa?

Wikipie-tan: Znalazłam 10 stron, oto kilka linków: *Antylopa*, *Antylopa szablroga*, *Antylopa końska*. Spróbuj doprecyzować czego dokładnie szukasz, to postaram się udzielić dokładniejszej odpowiedzi.

Chatbot informuje użytkownika, że istnieje dużo stron spełniających podane kryteria i zachęca do rozwinięcia myśli.

Ja: Antylopa gnu

Wikipie-tan: Strony Gnu pręgowane, *Antylopa*, *Krętorogie* mogą zawierać informację, której potrzebujesz.

Ja: Pa.

Wikipie-tan: Do zobaczenia.

Chatbot udziela precyzyjniejszej odpowiedzi. W tym przypadku użytkownik nie musiał podawać dodatkowych kryteriów, ponieważ szukana przez niego informacja znajdowała się na stronie wskazanej w pierwszej odpowiedzi.

Wynik wyszukiwania nie zawsze jest jednak trafny, przykład:

- Ja: Kto jest największym polskim poetą?

Wikipie-tan: Znalazłam 43 strony, oto kilka linków: *Polska*, *Luís de Camões*, *Kultura esperanta*. Spróbuj doprecyzować czego

dokładnie szukasz, to postaram się udzielić dokładniejszej odpowiedzi.

Ja: Romantyzmu.

Wikiped-tan: Znalazłam 25 stron, oto kilka linków: Synteza sztuk, Juliusz Słowacki, Jan Lechoń. Spróbuj doprecyzować czego dokładnie szukasz, to postaram się udzielić dokładniejszej odpowiedzi.

Odpowiedzi chatbota nie zaspokajają potrzeby informacyjnej rozmówcy. Taki wynik jest spowodowany częściowo tym, że Wikipedia nie powinna zawierać ocen – jest to sprzeczne z zasadą neutralnego punktu widzenia. Problem ten jest pogłębiany przez ograniczenia czasowe nakładane na chatbota, który powinien udzielić odpowiedzi możliwie szybko. Stworzenie własnego indeksu, który byłby zoptymalizowany do potrzeb aplikacji powinien być kolejnym krokiem w jej rozwoju.

### 4.3. Porównanie z innymi serwisami

Przyjęte założenia projektowe silnie uzależniają sprawne działanie aplikacji od zewnętrznych dostawców wyszukiwania. Aplikacja wykorzystuje te źródła w inny sposób niż człowiek, dlatego ma możliwość radzenia sobie lepiej w pewnych przypadkach. Jeśli poszukiwana strona nie znajdzie się w dostarczanych wynikach, to na pewno nie zostanie ona wzięta pod uwagę – ograniczenia dostawców wyszukiwania wpływają zatem na wyniki prezentowane przez aplikację.

Działanie aplikacji będzie porównywane z trzema serwisami: Google, Yahoo! Search oraz serwis KtoCo.pl. Wyszukiwarka [www.hipisek.pl](http://www.hipisek.pl) w chwili przeprowadzania porównania nie była dostępna, dlatego nie została wzięta pod uwagę.

Do każdej z wyszukiwarek zostały wysłane następujące zapytania:

- Gdzie znajduje się Poznań?
- Gdzie znajduje się cieśnina Naruto?
- Kto jest największym polskim poetą?
- Kto jest autorem Mazurka Dąbrowskiego?
- Gdzie występuje antylopa?
- W którym województwie znajduje się Kraków?
- Co to jest AGH?

Jako wyniki zostały uznane pierwsze 3 adresy stron wskazane przez każdą z wyszukiwarek. Linki sponsorowane nie były brane pod uwagę. Dla pierwszego zapytania *Gdzie znajduje się Poznań?* zostały rozpisanne szczegółowe wyniki, dane dla pozostałych są zebrane w tabeli 4.3.

- Opracowane rozwiązanie
  - Poznań – artykuł w Wikipedii na temat miasta
  - Poznań Główny – artykuł w Wikipedii na temat stacji kolejowej
  - Twierdza Poznań – artykuł w Wikipedii na temat fortyfikacji w Poznaniu

Pytanie	Opracowane rozwiązanie	Google	Yahoo	KtoCo.pl
Gdzie znajduje się Poznań?	3	0	2	0
Gdzie znajduje się cieśnina Naruto?	2	3	0	1
Kto jest największym polskim poetą?	0	3	0	1
Kto jest autorem Mazurka Dąbrowskiego?	3	2	2	1
Gdzie występuje antylopa?	3	3	3	3
W którym województwie znajduje się Kraków?	2	1	0	0
Co to jest AGH?	3	0	0	1

Tablica 4.1: Podsumowanie trafnych odpowiedzi udzielonych przez wyszukiwarki.

- Google

- hurtownia Czy ktoś wie gdzie jest w Poznaniu lub okolicy ... – wpis na forum rowerowym
- Ludzie chcą mieszkać tam, gdzie jest zielono – artykuł w serwisie [poznanzgazeta.pl](http://poznanzgazeta.pl)
- Gdzie jest Lech Poznań ? – artykuł o meczu piłkarskim

- Yahoo! Search

- Poznań – artykuł w Wikipedii na temat miasta
- Poznań – artykuł z Wikipedii na temat miasta umieszczony w mirrorze
- Poznań Dworzec Główny PKP – wpis na forum dotyczący zagospodarowania przestrzennego miasta

- KtoCo.pl

- Hotele Poznań - baza hoteli w Poznaniu – informacja o hotelach w mieście, link powtórzony trzy razy, z zaznaczonym opisem dla różnych hoteli

Patrząc na wyniki z tabeli, można stwierdzić, że wyszukiwarki ogólnego przeznaczenia nie radzą sobie dobrze z zapytaniami w języku naturalnym. Opracowane rozwiązanie dawało sobie radę z udzielaniem odpowiedzi. Należy jednak pamiętać, że chatbot przeszukiwał wyłącznie zasoby polskojęzycznej Wikipedii, na której szum informacyjny jest dużo niższy. W wynikach pozostałych wyszukiwarek pojawiały się m.in. filmy w serwisie youtube, wypowiedzi na forach, które nie niosły zbyt wiele treści. Patrząc na wyniki w tabeli można stwierdzić, że co najmniej jeden z dostawców zawsze wskazywał stronę z trafną odpowiedzią. Strategia polegająca na przejrzaniu stron wskazywanych przez różnych dostawców pozwala na zwiększenie szansy znalezienia właściwego dokumentu spełniającego potrzebę informacyjną.

## 5. Podsumowanie i wnioski

Celem niniejszej pracy było skonstruowanie chatbota, który wyszukuje informacje z określonej dziedziny wiedzy w zasobach polskojęzycznej Wikipedii. Wyszukiwanie miało odbywać się na podstawie zapytania sformułowanego przez użytkownika w języku naturalnym. Dla języka polskiego stanowi to wyzwanie, ponieważ rozbudowana fleksja oraz luźny szyk zdania znacznie utrudniają analizowanie wypowiedzi. Zadania nie ułatwia sposób, w jaki ludzie komunikują się ze sobą np. używając skrótów myślowych, czy wypowiedzi nie do końca poprawnych gramatycznie. Aby określić znaczenie danego przekazu trzeba mieć najczęściej informację o kontekście rozmowy, o samym rozmówcy lub też o otaczającym nas świecie.

Największym problemem podczas tworzenia systemu było znalezienie ogólnodostępnych materiałów o języku polskim, które zawierały wymagane dane, a jednocześnie warunki używania pozwalałyby na ich swobodne wykorzystanie. Bez odpowiednich narzędzi skonstruowanych na ich bazie poziom funkcjonowania całego rozwiązania byłby znacznie niższy. Stanowią one podstawę działania mechanizmów dopasowywania wzorców oraz klasyfikacji dokumentów umożliwiając analizę morfologiczną oraz znajdowanie synonimów wyrazów.

Stworzony system jest zgodny z przyjętymi na samym początku założeniami projektowymi. Do napisania pracy oraz stworzenia aplikacji wykorzystano wyłącznie oprogramowanie oraz biblioteki udostępniane na wolnych licencjach.<sup>1</sup>, przez co może ona być w całości rozpowszechniana i używana na podobnych zasadach. Z kolei modułowa budowa znacząco ułatwia zastępowanie i dodawanie nowych funkcjonalności – szkielet systemu może zostać użyty do napisania aplikacji spełniającej inne zadania.

Pomimo tego, iż projekt został zakończony sukcesem, nie stanowi on ostatecznego rozwiązania. Istnieje wiele obszarów, które z powodu ograniczonego czasu nie zostały ujęte w ramach tej pracy. Przykładowe usprawnienia, na które warto zwrócić uwagę to:

- automatyczne rozpoznawanie błędów pisowni i ich poprawianie<sup>2</sup>,
- zmiana sposobu działania maszyny stanowej, tak aby wybierana była tranzycja o największym współczynniku dopasowania, a nie pierwsza akceptująca wypowiedź użytkownika,
- cytowanie fragmentu strony zawierającego poszukiwaną informację lub odpowiadanie na pytanie poprzez parafrazę,
- przeszukiwanie zasobów siostrzanych projektów Wikipedii<sup>3</sup>,
- budowanie własnego indeksu zamiast wykorzystywać zewnętrzne wyszukiwarki,
- stworzenie wersji strony dla urządzeń mobilnych.

<sup>1</sup> Informacje na temat użytego oprogramowania zostały zawarte w dodatku A.

<sup>2</sup> Jednym z założeń projektowych pracy było traktowanie tekstu wprowadzonego przez użytkownika jako poprawnego.

<sup>3</sup> Więcej informacji na ten temat znajduje się w rozdziale 2.5.4.

## A. Wykorzystane biblioteki oraz oprogramowanie

Poniżej przedstawiona jest lista oprogramowania, które zostało wykorzystane do napisania niniejszej pracy<sup>1</sup>.

- System operacyjny – GNU/Linux, dystrybucja Gentoo 10.0
- Środowisko graficzne – K Desktop Environment 4
- Edytor grafiki – OpenOffice.org Draw 3.2
- Edytor tekstu – Kile 2.1
- Skład dokumentów - TeX Live 2008
- Edytor tekstu – Komodo Edit 5.2
- Nagrywanie płyt CD - K3b 2.0

Poniżej przedstawiona jest lista oprogramowania, które zostało wykorzystane do zaimplementowania rozwiązania, będącego częścią niniejszej pracy. W nawiasach zostały podane dokładne nazwy paczek oraz ich wersje, które zostały zainstalowane przy pomocy menedżera pakietów Portage oraz g-cpan.

- Serwer stron internetowych – **Apache** (www-servers/apache-2.2.15)
- Interpreter języka programowania – **Perl** (dev-lang/perl-5.10.1)
- Biblioteka do logowania komunikatów – **Log::Log4perl** (dev-perl/Log-Log4perl-1.28)
- Interfejs do logowania komunikatów – **Log::Any** (dev-perl/Log-Any-0.11)
- Biblioteka **Log::Any::Adapter::Log4perl** (perl-gcpan/Log-Any-Adapter-Log4perl-0.06)
- Biblioteka do programowania opartego na zdarzeniach – **AnyEvent** (dev-perl/AnyEvent-5.2.5.1)
- Biblioteka do asynchronicznej obsługi protokołu HTTP – **AnyEvent::HTTP** (perl-gcpan/AnyEvent-HTTP-1.45)
- Biblioteka do obsługi protokołu HTTP – **libwww-perl** (dev-perl/libwww-perl-5.836)
- Interfejs do przetwarzania danych w formacie JSON – **JSON::Any** (dev-perl/JSON-Any-1.22)
- Biblioteka do przetwarzania danych w formacie JSON – **JSON** (dev-perl/JSON-2.21)
- Interfejs do obsługi API MediaWiki – **MediaWiki::API**
- Biblioteka do przetwarzania danych w formacie RSS – **XML::RSS** (dev-perl/XML-RSS-1.48)
- Biblioteka **jQuery JavaScript Library** w wersji 1.4.2

---

<sup>1</sup>Pełna informacja na temat wykorzystanego oprogramowania oraz licencji znajduje się na nośniku dołączonym do pracy.

# Spis rysunków

2.1	Interfejs wyszukiwarki KtoCo.pl. . . . .	13
2.2	Interfejs wyszukiwarki KtoCo.pl, odpowiedź na zadane pytanie. . . . .	13
2.3	Interfejs wyszukiwarki Hipisek.pl. . . . .	14
2.4	Interfejs wyszukiwarki Hipisek.pl, odpowiedź na zadane pytanie. . . . .	14
2.5	Logo polskojęzycznej Wikipedii. . . . .	15
3.1	Środowisko wykonania aplikacji internetowej. . . . .	20
3.2	Logiczny podział stworzonej aplikacji. . . . .	21
3.3	Przykładowy diagram stanów . . . . .	27
3.4	Relacje między wybranymi komponentami. . . . .	33
3.5	Uproszczony schemat działania aplikacji. . . . .	37
4.1	Wikipie-tan – maskotka Wikipedii. . . . .	39
4.2	Interfejs wyszukiwarki – rozpoczęcie konwersacji. . . . .	39
4.3	Prowadzenie konwersacji. . . . .	40
4.4	Proste wyszukiwanie. . . . .	41
4.5	Ujednoznacznienie wyszukiwania. . . . .	43

# Spis tablic

3.1	Przykładowe słowa określające rodzaj poszukiwanej informacji. . . . .	34
3.2	Wpływ odmiany czasownika na oczekiwane rezultaty wyszukiwania. . . . .	34
3.3	Przykładowe przyporządkowanie wag poszczególnym klasom słów. . . . .	36
4.1	Podsumowanie trafnych odpowiedzi udzielonych przez wyszukiwarki. . . . .	45



## Bibliografia

- [1] W. Abramowicz. *Filtrowanie informacji*. Wydawnictwo Akademii Ekonomicznej w Poznaniu, Poznań, 2008.
- [2] J. Daciuk. *Incremental Construction of Finite-State Automata and Transducers, and their Use in the Natural Language Processing*. Gdańsk, 1998.
- [3] M. A. Kłopotek. *Inteligentne wyszukiwarki internetowe*. Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2001.
- [4] K. Polański, editor. *Encyklopedia językoznawstwa ogólnego*. Zakład Narodowy im. Ossolińskich, Wrocław, 1993.
- [5] Stanusch Technologies. KtoCo.pl - polska wyszukiwarka semantyczna. [http://www.stanusch.com/#subpages/produkty/wiper.html?menu=subpages/produkty/menu\\_wiper.html](http://www.stanusch.com/#subpages/produkty/wiper.html?menu=subpages/produkty/menu_wiper.html), 2010. [dostęp: 2010-03-10].
- [6] R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *J. ACM*, 21(1):168–173, 1974.
- [7] M. Walas. Hipisek od kuchni. <http://re-research.wmi.amu.edu.pl/node/30>, 2009.
- [8] M. Walas. Pozyskiwanie wiedzy z bazy artykułów/tekstów za pomocą aplikacji typu wyszukiwarki internetowej odpowiadającej na pytania o czas i miejsce. Master's thesis, Uniwersytet im. Adama Mickiewicza, Poznań, 2009.
- [9] L. Wall, T. Christiansen, and J. Orwant. *Perl – programowanie, wydanie drugie*. Wydawnictwo RM, Warszawa, 2001.
- [10] Wikipedia. Chatterbot. <http://en.wikipedia.org/wiki/Chatterbot?oldid=344755198>, 2010. [dostęp: 2010-03-08].
- [11] Wikipedia. Edit distance. [http://en.wikipedia.org/wiki/Edit\\_distance?oldid=308424443](http://en.wikipedia.org/wiki/Edit_distance?oldid=308424443), 2010. [dostęp: 2010-08-21].
- [12] Wikipedia. Finite-state machine. [http://en.wikipedia.org/wiki/Finite-state\\_machine?oldid=354971263](http://en.wikipedia.org/wiki/Finite-state_machine?oldid=354971263), 2010. [dostęp: 2010-04-13].
- [13] Wikipedia. List of wikipedias. [http://meta.wikimedia.org/wiki/List\\_of\\_Wikipedias?oldid=898426](http://meta.wikimedia.org/wiki/List_of_Wikipedias?oldid=898426), 2010. [dostęp: 2010-03-03].
- [14] Wikipedia. Przetwarzanie języka naturalnego. [http://pl.wikipedia.org/wiki/Przetwarzanie\\_języka\\_naturalnego?oldid=20548025](http://pl.wikipedia.org/wiki/Przetwarzanie_języka_naturalnego?oldid=20548025), 2010. [dostęp: 2010-03-01].

- 
- [15] Wikipedia. Wikipedia. <http://en.wikipedia.org/wiki/Wikipedia?oldid=353396975>, 2010. [dostęp: 2010-04-01].
- [16] Wikipedia. Wikipedia:Pięć filarów. [http://pl.wikipedia.org/wiki/Wikipedia:Pięć\\_filarów?oldid=20681236](http://pl.wikipedia.org/wiki/Wikipedia:Pięć_filarów?oldid=20681236), 2010. [dostęp: 2010-03-21].
- [17] Wolfram Alpha LLC. About WolframAlpha. <http://www.wolframalpha.com/about.html>, 2010. [dostęp: 2010-05-08].