WPROWADZENIE DO CLIPS'a 6.x

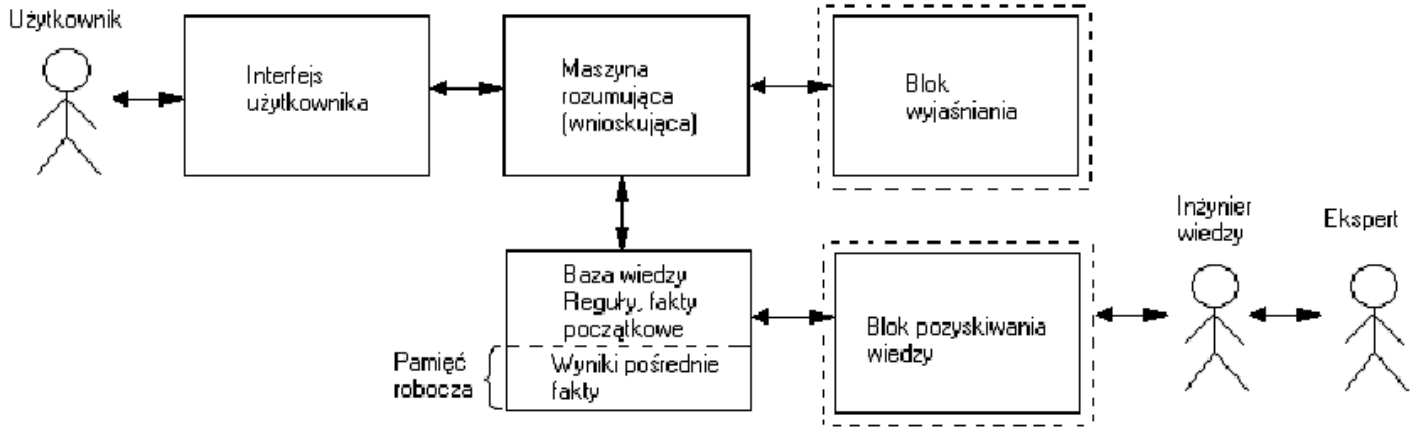
1. Wprowadzenie

CLIPS jest specjalizowanym językiem programowania, przeznaczonym do tworzenia systemów ekspertowych. Został on opracowany przez NASA/Johnson Space Center i upowszechniony w roku 1988 (wersja 3.2).

System ekspertowy jest „inteligentnym" programem komputerowym tak skonstruowanym, aby mógł naśladować postępowanie człowieka-eksperta w pewnej dziedzinie wiedzy przy rozwiązywaniu problemów z tej dziedziny. System taki musi więc zawierać wiedzę eksperta zakodowaną w postaci reguł typu:
JEŻELI ... TO ... oraz procedurę wnioskowania. Systemy ekspertowe stosuje się zwłaszcza w dziedzinach, które nie są w jednoznaczny sposób sformalizowane, a więc gdy nie istnieje algorytm rozwiązania postawionego problemu. Do rozwiązania problemu niezbędna jest wtedy wiedza i doświadczenie eksperta. Właśnie owo doświadczenie, zdobyte w trakcie wieloletniej działalności eksperta, pozwala na uzyskiwanie wysokiej jakości rozwiązań, w stosunkowo krótkim czasie, dzięki stosowaniu rozumowania heurystycznego. Rozumowanie heurystyczne polega na wykorzystaniu intuicji, zdobytych doświadczeń, „skrótów myślowych" itp., co stanowi uzupełnienie sformalizowanej wiedzy z danej dziedziny.
Typowe zastosowania systemów ekspertowych to:

- klasyfikacja – przyporządkowanie obiektów do danych grup ze względu na ich właściwości,
- interpretacja – opis sytuacji wnioskowany np. z pomiarów przy użyciu dużej liczby czujników, przyrządów pomiarowych lub innych danych,
- prognozowanie – opis przewidywanych konsekwencji, wywnioskowanych z danej sytuacji (np. przewidywanie zmian kursów walut, wskaźników giełdowych itd.),
- diagnozowanie – zarówno medyczne, jak i techniczne (urządzeń, systemów) – wnioskowanie o chorobie (lub uszkodzeniach) badanego pacjenta (lub systemu) na podstawie obserwacji bądź pomiarów,
- terapia – określenie czynności, które należy wykonać w celu usunięcia niedomagań określonych przez diagnozę,
- monitorowanie – nadzorowanie przebiegu procesu i porównanie z przebiegiem poprawnym,
- sterowanie – obiektami lub procesami – zwłaszcza systemy ekspertowe z logiką rozmytą,
- konfigurowanie – składanie urządzenia z podzespołów w sposób, który ma zapewnić spełnienie założonych wymagań (np. żądań klienta),
- projektowanie – podobnie jak konfigurowanie, lecz na poziomie podstawowych elementów (np. projektowanie wzmacniacza z tranzystorów, diod, rezystorów itd.)

Struktura systemu ekspertowego jest inna niż programu konwencjonalnego, gdyż można w niej wydzielić osobne autonomiczne bloki programowe, takie jak baza wiedzy i blok rozumowania (maszyna wnioskująca) oraz bloki pomocnicze jak sprzęg (ang. Interface) z użytkownikiem oraz blok wyjaśniania. Na rys. 1 przedstawiono schemat blokowy systemu ekspertowego.



Rys. 1 – Schemat blokowy systemu ekspertowego

Sprzęg (Interface) służy do dogodnego porozumiewania się użytkownika z komputerem w sposób zbliżony do języka naturalnego, z ograniczonym zasobem słów i uproszczoną składnią. W chwili obecnej najczęściej stosuje się komunikację poprzez klawiaturę i monitor. W realizacji „przyjacielskiego" sprzęgu zwraca się

szczególną uwagę na to aby użytkownik, nie znając zasad programowania, prawie natychmiast po zetknięciu się z systemem ekspertowym (komputerem) mógł z niego korzystać. Zapewnia się to przede wszystkim przez programowe zapewnienie dialogu komputera z użytkownikiem w języku zbliżonym do naturalnego, przy czym na początku dialogu inicjatywę przejmuje komputer wyjaśniając na ekranie monitora co należy uczynić, aby zainicjować i prowadzić dialog. Następnie komputer zadaje pytania użytkownikowi w celu uzyskania niezbędnych danych o problemie do rozwiązania aby móc zainicjować proces rozumowania. Możliwość wyboru opcji przedstawianych przez system użytkownikowi podawana jest bardzo często w postaci wielopoziomowego menu.

Baza wiedzy zawiera fakty (dane) opisujące obiekty lub zdarzenia dotyczące danego problemu, oraz wiedzę ogólną dotyczącą danej dziedziny obejmującej dany problem do rozwiązania. Wiedza ta ujmująca związki miedzy obiektami i zdarzeniami jest zapisywana najczęściej w postaci reguł. Wiedza zapisywana jest głównie w sposób deklaratywny, tj. opisowy, bez wskazywania sekwencji w jakiej jej fragmenty powinny być wykonywane, wykorzystywane lub przetwarzane. W zależności od pochodzenia, wiedza może dotyczyć zagadnień ogólnych w obrębie wąskiej dziedziny i heurystycznych sposobów rozwiązywania problemów i być pozyskana od eksperta, lub dotyczyć szczegółów danego problemu do rozwiązania i pochodzić od użytkownika.

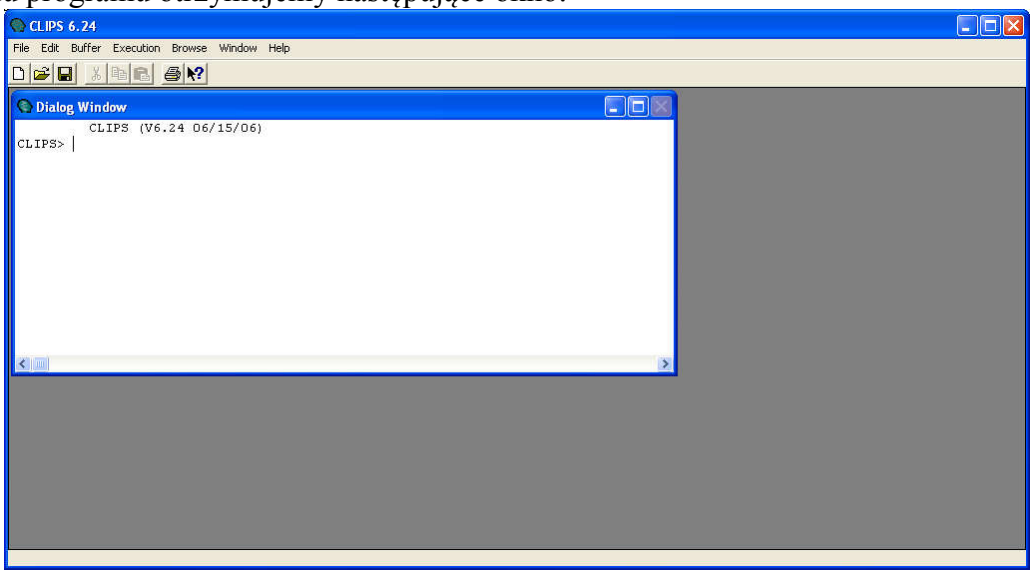
Blok rozumowania, nazywany maszyną wnioskującą jest algorytmiczną częścią sterującą wykonywaniem programu. W przypadku zapisu wiedzy w postaci reguł, blok ten nazywany jest interpretatorem reguł, gdyż powoduje on przeglądanie zestawu reguł i wykonywanie tych, które odpowiadają faktom istniejącym w bazie wiedzy lub wprowadzonym przez użutkownika.

Blok wyjaśniania umożliwia odpowiedzi na ewentualne pytania użytkownika odnośnie sposobu dojścia systemu ekspertowego do konkluzji, bądź wyjaśnienie dlaczego system ekspertowy żąda od użytkownika wprowadzenia dodatkowych danych. Nie jest on niezbędny do działania systemu ekspertowego, lecz jego obecność znacznie ułatwia posługiwanie się nim i wyraźnie podnosi jego jakość i wiarygodność dochodzenia do konkluzji przy rozwiązywaniu problemów.

2. Obsługa programu CLIPS

Proszę pobrać program CLIPS z adresu http://aslowik.5v.pl/CLIPS/CLIPSWin.exe

Po uruchomieniu programu otrzymujemy następujące okno:



Rys. 2 – Okno główne programu CLIPS

Aby zobaczyć przykładowy system ekspertowy w działaniu należy zgrać przykładowe programy znajdujące się pod adresem:

Prosty system ekspertowy do sprawdzania grzybów:

http://aslowik.5v.pl/CLIPS/Grzyby/Grzyby.clp

lub

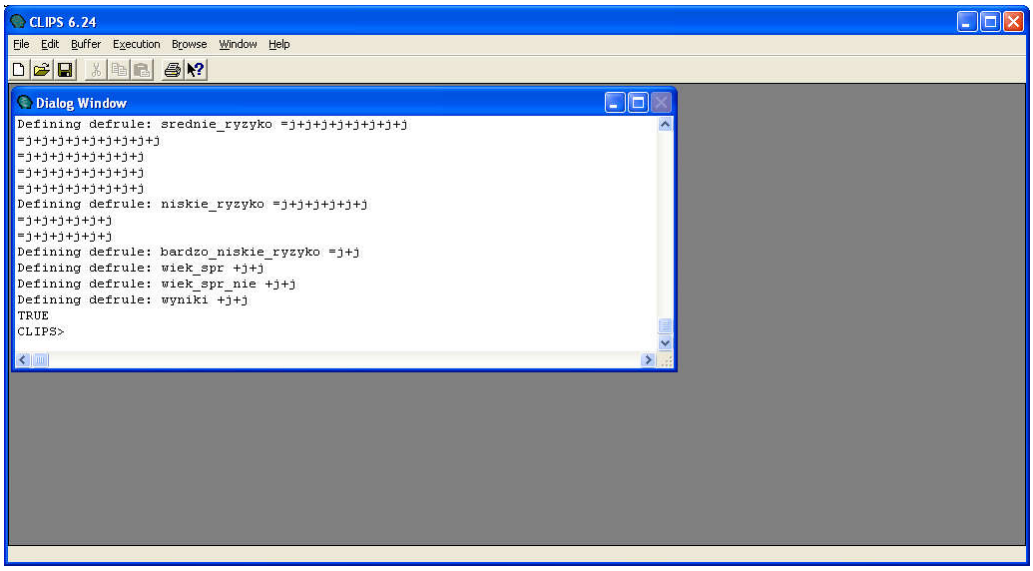
Program do diagnostyki opon

http://aslowik.5v.pl/CLIPS/Opony/Opony.clp

Po zapisaniu programów na dysku, aby je uruchomić należy w CLIPS'ie wydać następujące polecenia:

- Execution -> Clear CLIPS
  (czyści przestrzeń roboczą CLIPS'a łącznie z bazą wiedzy)
- File -> Load
  (i tu wskazujemy który plik chcemy wczytać) – w tym miejscu program CLIPS'owy jest wprowadzany do pamięci

Jeśli wczytanie programu przebiegło pomyślnie to powinniśmy otrzymać na końcu komunikat TRUE, tak jak na poniższym rysunku.



Rys. 3 – Okno po wczytaniu przykładowego programu

Wyświetlona wartość TRUE oznacza, że program jest napisany bez błędów w składni. W przypadku, gdy w oknie zamiast TRUE pojawi się wartość FALSE, oznacza to, że program posiada błędy i należy je skorygować.

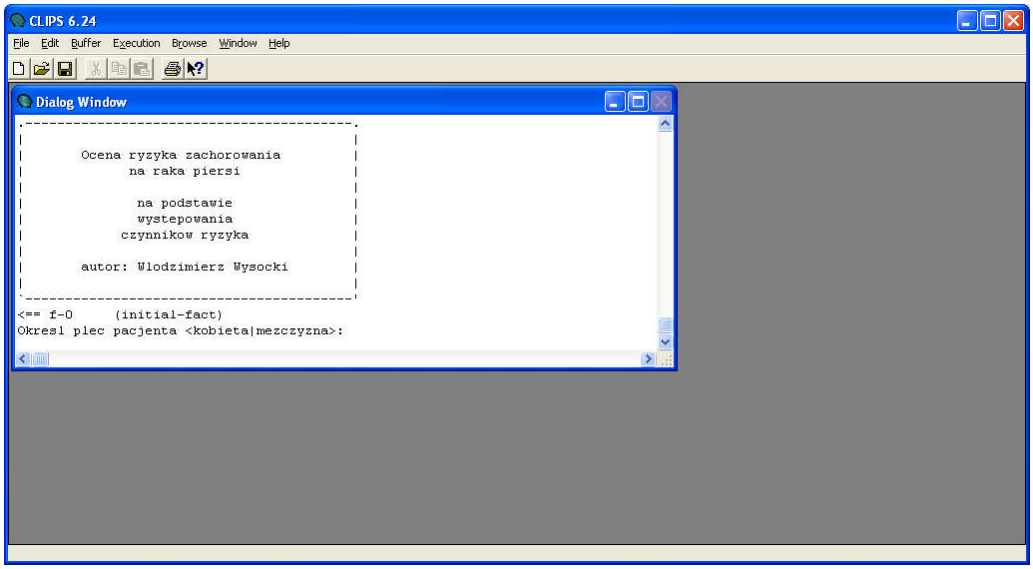
Jeśli w oknie mamy wartość TRUE wówczas możemy uruchomić program w tym celu należy wybrać:

- Execution -> Reset (wprowadza fakt początkowy tzw. „initial-fact" do pamięci komputera)

a następnie

- Execution -> Run

wynikiem działania tych komend powinien być poniższy ekran (w przypadku wczytania pliku „projekt2.clp")



Rys. 4 – Uruchomiony przykładowy program

Teraz program prowadzi dialog z użytkownikiem w celu wystawienia poprawnej diagnozy.

3. Jak napisać własny program w środowisku CLIPS ?

W tym celu wybieramy:
- File -> New (jeśli tworzymy plik od początku)

lub

- File -> Open (jeśli otwieramy już istniejący plik do edycji)

Najprostszy program w CLIPS'ie będzie wyglądał następująco:

```
(defrule R1
(initial-fact)
=>
(printout t „Hello world !” crlf)
)
```

Teraz wystarczy go zapisać z rozszerzeniem *.clp, wczytać i uruchomić).

4. Podstawowe typy danych w CLIPS'ie

Każdy program składa się z reguł, gdyż dowolna wiedza może być w formie reguł zapisana. Np.

IF Dzis=niedziela AND Pora_Dnia=rano THEN są zajęcia z Hybryd.

W CLIPS'ie każda reguła zapisywana jest następująco:

```
(defrule nazwa_reguły
(przesłanka_1)
...
(przesłanka_n)
=>
(akcja_1)
...
(akcja_n))
```

Pożyteczne komendy (stosowane tylko po prawej stronie reguły za znacznikiem „=>”):

bind – przypisanie danych do zmiennej

Np. (bind ?x ?y) <- do zmiennej ?x przypisz wartość zmiennej ?y
(bind ?x 2) <- do zmiennej ?x przypisz wartość 2
(bind ?xyz (read)) <- do zmiennej ?xyz przypisz wartość wprowadzoną z klawiatury

printout t – wyświetlanie tekstu na ekranie

Np. (printout t „Hello world !”) <- wyświetlenie napisu “Hello world !”
(printout t “x=” ?aaa) <- wyświetlenie wartości zmiennej ?aaa np. „x=5” jesli ?aaa ma wartość 5
(printout t „Ala ma kota” crlf) <- wyświetlenie tekstu „Ala ma kota” i przejście do nowego wiersza
crlf – oznacza przejście do nowego wiersza

assert – potwierdzenie nowego faktu

Np. (assert (wiek 15)) <- dodanie nowego faktu (wiek 15)
(assert (wiek ?x)) <- jeśli w zmiennej ?x znajduje się np. wartość 2, to zostanie dodany fakt (wiek 2)

Przykład:
Załóżmy, że chcemy napisać program, który w regule R1 pobierze od użytkownika jego imię, a następnie wyświetli je na ekranie w regule R2.

Tworzymy regułę R1:

```
(defrule R1
(initial-fact)   ; fakt początkowy – zawsze musi zaistnieć w regule początkowej
=>
(printout t „Podaj imię:” crlf)     ;wypisanie tekstu na ekran
(bind ?imie (read))                 ;pobranie danych z klawiatury i wpisanie ich do zmiennej ?imie
(assert (imie ?imie))               ; utworzenie faktu (imie „nazwa_ktora_podal_uzytkownik”)
)

(defrule R2
(imie ?imie)   ; reguła R2 uruchamia się gdy zaistnieje fakt (imie o dowolnej wartości do której dostęp mamy
               ; pod zmienna ?imie
=>
(printout t „Witaj” ?imie „ na zajeciach z Hybryd” crlf)
)
```

5. Zadania do wykonania
a). uruchomić program CLIPS
b). wczytać i uruchomić przykładowe projekty:

http://aslowik.5v.pl/CLIPS/Grzyby/Grzyby.clp
http://aslowik.5v.pl/CLIPS/Opony/Opony.clp

c). napisać program złożony z 3 reguł:

- Reguła R1 uruchamia się na fakt początkowy (initial-fact), w regule tej podajemy z klawiatury imię użytkownika i na tej podstawie tworzymy fakt (imie ?imie).

- Reguła R2 uruchamia się jeśli zaistnieje fakt (imie ?imie) i prosi o podanie nazwiska, na podstawie którego tworzy fakt (nazwisko ?nazwisko)

- Reguła R3 uruchamia się jeśli zaistnieje fakt (imie ?imie) oraz (nazwisko ?nazwisko) i wypisuje na ekranie komunikat przykładowej postaci:

„Nazywasz sie Jan Nowak”

d). napisać program złożony z 4 reguł:

- Reguła R1 uruchamia się na fakt początkowy (initial-fact), w regule tej podajemy z klawiatury imię użytkownika i na tej podstawie tworzymy fakt (imie ?imie).

- Reguła R2 uruchamia się jeśli zaistnieje fakt (imie ?imie) i prosi o podanie nazwiska, na podstawie którego tworzy fakt (nazwisko ?nazwisko)

- Reguła R3 uruchamia się jeśli zaistnieje fakt (imie ?imie) oraz (nazwisko ?nazwisko) i wprowadzamy wiek, oraz tworzymy odpowiedni fakt (wiek ?wiek)

- Reguła R4 uruchamia się jeśli zaistnieje fakt (imie ?imie), (nazwisko ?nazwisko) oraz (wiek ?wiek) i wypisuje na ekranie komunikat przykładowej postaci:

”Nazywasz sie Jan Nowak i masz 37 lat”