**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Новгородский государственный университет имени Ярослава Мудрого»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра ИТиС

Отчёт по

лабораторной работе

«ПРОСТЕЙШИЙ ИНТЕРФЕЙС НА ЕСТЕСТВЕННОМ ЯЗЫКЕ С ИСПОЛЬЗОВАНИЕМ КЛЮЧЕВЫХ СЛОВ»

дисциплина:

«Системы искусственного интеллекта»

Выполнил: ст. гр. 9091

\_\_\_\_\_\_\_ \_\_\_\_Мирошкин О.А.

Проверил: Преподаватель

\_\_\_\_\_\_\_\_\_\_\_Михайлов Д.В.

Великий Новгород

2022

# Формулировка цели и задач

Целью работы является изучение принципов построения и организации экспертных систем, базирующихся на логике и правилах.

# Данные, использованные в ЭС

Предметная область для ЭС – ткани.

Ткани я разбил на группы в зависимости от типа исходного сырья: натуральный, искусственный, синтетический.

Таблица 1. Описание предметной области

|  |  |  |
| --- | --- | --- |
| **Группа** | **Название ткани** | **Свойства** |
| **Натуральный** | **Хлопок** | Высокое качество  Дороговизна  Хорошая прочность |
| **Лён** | Высокое качество  Дороговизна  Хорошая воздухопроницаемость |
| **Шерсть** | Высокое качество  Дороговизна  Антиаллергенность |
| **Искусственный** | **Бамбук** | Дешевизна  Хорошая воздухопроницаемость  Хорошая прочность |
| **Ацетат** | Дешевизна  Хорошая воздухопроницаемость  Хорошая терморегуляция |
| **Синтетический** | **Акрил** | Дешевизна  Хорошая терморегуляция  Антиаллергенность |

1. **Описание характеристик разработанных ЭС**

Наиболее известными являются следующие подходы к решению

задач ЕЯ-общения конечного пользователя с ЭВМ:

— Прагматический анализ - наиболее сложный, связан с изучением смысла предложения в связи с внеязыковой действительностью. Анализ ключевых слов — метод анализа ЕЯ-высказываний на предмет наличия ключевых слов, которые становятся значениями аргументов предикатов. При этом компьютер одинаково реагирует на различные варианты входного текста, наличие грамматической правильности предложений не является обязательным, роль играет лишь наличие ключевых слов. Применение -построение ЕЯ-интерфейсов к Базам Данных (БД).

— Грамматический анализ: контекстно-свободный и контекстно-зависимый. Контекстно-Свободный (КС) анализ — ЕЯ-фразы классифицируются в зависимости от их внутренней структуры вне зависимости от контекста в соответствии с грамматическими правилами, задающими порядок следования допустимых языком символов (слов). Здесь следует отметить синтаксический анализ предложений.

— Прагматический анализ — наиболее сложный, связан с изучением смысла предложения с учетом связи с внеязыковой действительностью.

В этой лабораторной интерфейс на естественном языке реализован с помощью ключевых слов. Введённая фраза разбивается на слова, после чего из этих слов выбираются перовое и последнее. Если первое слово начинается на “вых” или “вый” независимо от второго слова происходит выход из программы. Если первое слово начинается на “най”, а последнее слово – “ткань” делается вывод что пользователь хочет найти вирус и запускается процесс поиска по БЗ. Если ни одно из предыдущих условий не выполнилось пользователь получает сообщение об ошибке в формате “ Введённые вами ключевые слова ПЕРВОЕ\_СЛОВО\_ИЗ\_ФРАЗЫ и ПОСЛЕДНЕЕ\_СЛОВО\_ИЗ\_ФРАЗЫ не известны системе.”.

# Тестовые данные

Подается на ввод: Выйти из программы

Результат: Программа завершает свою работу.

Подается на ввод: Бессмысленный запрос

Результат: Введённые вами ключевые слова бессмысленный и запрос не известны системе.

Подается на ввод: Найти ткань  
Входные данные: Да, Да, Да, Да

Результат: Предполагаемая ткань: Хлопок.

Подается на ввод: Найди ткань

Входные данные: Нет, Да, Да, Да, Да

Результат: Предполагаемая ткань: Бамбук.

Подается на ввод: Найди мне пожалуйста ткань

Входные данные: Да, Да, Да, Нет, Да

Результат: Предполагаемая ткань: Лён.

1. **Вывод**

Я на практике научился использовать лингвистические возможности языка Пролог для построения интерфейса на естественном языке (пускай и в максимально примитивной форме) для взаимодействия с пользователем.

# Приложение

## Текст программы

Файл logic\_es\_win.pro

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright (c) НовГУ

Project: LOGIC\_ES\_WIN

FileName: LOGIC\_ES\_WIN.PRO

Purpose: No description

Comments:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

include "logic\_es\_win.inc"

include "logic\_es\_win.con"

include "hlptopic.con"

%BEGIN\_WIN Task Window

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Event handling for Task Window

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

predicates

task\_win\_eh : EHANDLER

rev(word\_list,word\_list,word\_list).

reverse(word\_list,word\_list).

nondeterm rule(rule\_number,category,category,conditions)

nondeterm cond(cond\_number,condition)

nondeterm topic(condition)

assert\_database

do\_consulting

% nondeterm info

erase

clear

do\_answer(cond\_number,integer)

% keyword(category)

do\_right\_form(string,category)

func\_keyword(string)

last(word\_list, string)

member(char,char\_list).

nondeterm symbol\_counter(string,integer).

nondeterm del\_front\_space(string,string).

nondeterm fronttoken\_cyr(string,string,string).

nondeterm convers(string,word\_list)

nondeterm upper\_lower\_cyr(string,string).

nondeterm upper\_lower\_cyr\_convers(char\_list,char\_list).

nondeterm str\_char\_list(string,char\_list).

pack(char\_list,string).

goes(string,category)

nondeterm go(history,category)

nondeterm check(rule\_number,history,conditions)

ask\_question(cond\_number,condition)

constants

%BEGIN Task Window, CreateParms, 21:26:22-22.12.2007, Code automatically updated!

task\_win\_Flags = [wsf\_SizeBorder,wsf\_TitleBar,wsf\_Close,wsf\_Maximize,wsf\_Minimize,wsf\_ClipSiblings]

task\_win\_Menu = res\_menu(idr\_task\_menu)

task\_win\_Title = "Экспертная система, базирующаяся на логике"

task\_win\_Help = idh\_contents

%END Task Window, CreateParms

clauses

/\* База знаний. \*/

/\* Размещение в резидентной БД информации из утверждений БЗ ЭС \*/

assert\_database:-

rule(Rule\_number,Category,Type\_of\_Sport,Conditions),

assertz(d\_rule(Rule\_number,Category,Type\_of\_Sport,Conditions)),fail.

assert\_database:-

cond(Cond\_number,Condition),

assertz(d\_cond(Cond\_number,Condition)),fail.

assert\_database:-

topic(Condition),

assertz(d\_topic(Condition)),fail.

assert\_database:-!.

/\* Условия-характеристики различных тканей.\*/

cond(1,"натуральный тип сырья").

cond(2,"искусственный тип сырья").

cond(3,"синтетический тип сырья").

cond(4,"обладает антиаллергенность").

cond(5,"имеет хорошая прочность").

cond(6,"обладает нерастягиваемость").

cond(7,"обладает высокое качество").

cond(8,"обладает дороговизна").

cond(9,"обладает дешевизна").

cond(10,"имеет хорошая воздухопроницаемость").

cond(11,"имеет хорошая терморегуляция").

cond(12,"имеет хорошая влагопоглощаемость").

/\* Данные о типах ткани \*/

topic("натуральная").

topic("искусственная").

topic("синтетическая").

/\* Данные о конкретных вирусах \*/

rule(1,"ткань","натуральная",[1]).

rule(2,"ткань","искусственная",[2]).

rule(3,"ткань","синтетическая",[3]).

rule(3,"натуральная","Хлопок",[7,8,5]).

rule(4,"натуральная","Лён",[7,8,10]).

rule(5,"натуральная","Шерсть",[7,8,4]).

rule(6,"искусственная","Бамбук",[9,10,5]).

rule(7,"искусственная","Ацетат",[9,10,11]).

rule(8,"синтетическая","Акрил",[9,10,4]).

do\_consulting:-

goes(\_,\_),!.

% go([],First\_keyword),!.

do\_consulting:-

not(dummy),

dlg\_Error("Информация об интересующей вас ткани отсутствует в БЗ."),

clear.

/\* Запрос и получение ответов yes и no от пользователя \*/

ask\_question(Sport\_cond,Text):-

concat("Вопрос : ",Text,Temp),

concat(Temp," ",Temp1),

concat(Temp1,"?",Quest),

Response1=dlg\_Ask("Консультация",Quest,["Да","Нет"]),

Response=Response1+1,

do\_answer(Sport\_cond,Response).

/\* Предикаты ЕЯ-интерфейса \*/

/\* Реверсирование списка \*/

rev([],Init,Init).

rev([H|T],Init,Res):-

rev(T,[H|Init],Res).

reverse(Arg,Res):-

rev(Arg,[],Res).

/\* Принадлежность элемента списку \*/

member(Head,[Head|\_]):-!.

member(Elem,[\_|T]):-

member(Elem,T).

/\* Подсчет символов в строке до конца строки, либо ближайшего пробела,

символа возврата каретки, перевода строки, !,",#,$ \*/

symbol\_counter(Str,0):-

frontchar(Str,'\32',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\10',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\13',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\33',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\34',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\35',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\36',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\40',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\41',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\44',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\45',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\46',\_),!.

symbol\_counter(Str,0):-

frontchar(Str,'\59',\_),!.

symbol\_counter("",0).

symbol\_counter(Str,Number):-

frontchar(Str,\_Char,Rest\_of\_string),

symbol\_counter(Rest\_of\_string,Number1),

Number=Number1+1.

/\* Выделение подстроки до первого разделителя \*/

fronttoken\_cyr(Str,Token,Rest\_of\_string):-

symbol\_counter(Str,Number),

frontstr(Number,Str,Token,Rest\_of\_string).

/\* Удаление разделителя в начале строки \*/

del\_front\_space("","").

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\32',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\10',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\13',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\33',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\34',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\35',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\36',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\44',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\40',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\41',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\45',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\46',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Res):-

frontchar(Arg,Char,Res1),Char='\59',!,

del\_front\_space(Res1,Res).

del\_front\_space(Arg,Arg):-frontchar(Arg,Char,\_),

not(member(Char,['\32','\10','\13','\33',

'\34','\35','\36','\40','\41',

'\44','\45','\46','\59'])).

/\* Модифицированное правило преобразования строки в список

слов \*/

convers("",[]):-!.

convers(Str,[Head1|Tail]):-

fronttoken\_cyr(Str,Head,Str2),

upper\_lower\_cyr(Head,Head1),

del\_front\_space(Str2,Str1),

convers(Str1,Tail).

/\* Предикат upper\_lower для кирилицы Windows \*/

upper\_lower\_cyr(InString,OutString):-

str\_char\_list(InString,Char\_List\_for\_InString),

upper\_lower\_cyr\_convers(Char\_List\_for\_InString,Char\_List\_for\_OutString),

pack(Char\_List\_for\_OutString,OutString).

upper\_lower\_cyr\_convers([],[]).

upper\_lower\_cyr\_convers([Char|Char\_List],[Char1|Char\_List1]):-

char\_int(Char,ASCII\_code),

ASCII\_code>=192,ASCII\_code<=223,!,

ASCII\_code\_new=ASCII\_code+32,

char\_int(Char1,ASCII\_code\_new),

upper\_lower\_cyr\_convers(Char\_List,Char\_List1).

upper\_lower\_cyr\_convers([Char|Char\_List],[Char1|Char\_List1]):-

char\_int(Char,ASCII\_code),

ASCII\_code=168,!,

ASCII\_code\_new=ASCII\_code+16,

char\_int(Char1,ASCII\_code\_new),

upper\_lower\_cyr\_convers(Char\_List,Char\_List1).

upper\_lower\_cyr\_convers([Char|Char\_List],[Char1|Char\_List1]):-

char\_int(Char,ASCII\_code),

ASCII\_code>=65,ASCII\_code<=90,!,

ASCII\_code\_new=ASCII\_code+32,

char\_int(Char1,ASCII\_code\_new),

upper\_lower\_cyr\_convers(Char\_List,Char\_List1).

upper\_lower\_cyr\_convers([Char|Char\_List],[Char|Char\_List1]):-

upper\_lower\_cyr\_convers(Char\_List,Char\_List1).

/\* Преобразование строки в список символов \*/

str\_char\_list("",[]).

str\_char\_list(Word,[Char|Char\_List]):-

frontchar(Word,Char,WordRest),

str\_char\_list(WordRest,Char\_List).

/\* Превращение списка символов в строку \*/

pack([],"").

pack([H|T],Res):-

str\_char(Str\_H,H),

pack(T,Res1),

concat(Str\_H,Res1,Res).

/\* Проверка правильности ключевого слова \*/

% keyword(Keyword):-

% rule(\_,Keyword,\_,\_),!.

% keyword(Keyword):-

% rule(\_,\_,Keyword,\_),!.

/\* Правило для проверки ключевых слов \*/

do\_right\_form (Kname, Lname):-

func\_keyword (Kname),

rule (\_,Lname,\_,\_),

go([],Lname), !.

do\_right\_form (Kname,\_):-

frontstr (3, Kname, Word, \_),

upper\_lower\_cyr (Word, Key),

Key="вый",!,

exit.

do\_right\_form (Kname,\_):-

frontstr (3, Kname, Word, \_),

upper\_lower\_cyr (Word, Key),

Key="вых",!,

exit.

do\_right\_form (Kname,\_):- upper\_lower\_cyr (Kname, Word),

Word="exit", !,

exit.

do\_right\_form (Kname,\_):-

upper\_lower\_cyr (Kname, Word), Word="quit", !,

exit.

do\_right\_form (Kname, Lname):-

concat ("Введенные вами ключевые слова : ", Kname, Temp),

concat (Temp, " и ",Temp1),

concat (Temp1, Lname, Temp2),

concat (Temp2, " не известны системе. ", Msg),

dlg\_Note (Msg), !.

/\* Пример проверки допустимости ключевого слова-приказа \*/

func\_keyword (Word):-

frontstr (3, Word, Key, \_),

upper\_lower\_cyr (Key, KeyTr),

KeyTr="най", !.

/\* Поиск первого ключевого слова в высказывании пользователя \*/

% first\_keyword\_in\_sentence([Head|\_],Head):-

% keyword(Head),!.

% first\_keyword\_in\_sentence([\_|Tail],First\_keyword):-

% first\_keyword\_in\_sentence(Tail,First\_keyword).

% first\_keyword\_in\_sentence([],\_):-!,info,fail.

/\* Последний элемент списка \*/

last([X], X):-!.

last([\_Head|Tail], Element):-

last(Tail, Element).

/\* Ввод запроса на естественном (русском) языке \*/

goes(Mygoal,First\_keyword):-

Mygoal=dlg\_GetStr("Консультация","Введите Ваш запрос : ","Текст вашего запроса"),

convers(Mygoal,Word\_list),

/\* do\_right\_form (Word\_list, First\_keyword), \*/

last(Word\_list, Lname),

last(Word\_list, First\_keyword),

reverse(Word\_list, RWrod\_list),

last(RWrod\_list, Kname),

do\_right\_form(Kname, Lname),!.

% first\_keyword\_in\_sentence(Word\_list,First\_keyword),!.

/\* Механизм вывода \*/

/\* Начальное правило механизма вывода \*/

go(\_,Mygoal):-

not(rule(\_,Mygoal,\_,\_)),!,

concat("Предполагаемая ткань : ",Mygoal,Temp),

concat(Temp,".",Result),

dlg\_Note("Экспертное заключение : ",Result).

go(History,Mygoal):-

rule(Rule\_number,Mygoal,Type\_of\_Sport,Conditions),

check(Rule\_number,History,Conditions),

go([Rule\_number|History],Type\_of\_Sport).

/\* Сопоставление входных данных пользователя со списками атрибутов

отдельных видах спорта \*/

check(Rule\_number,History,[Sport\_cond|Rest\_Sport\_cond\_list]):-

yes(Sport\_cond),!,

check(Rule\_number,History,Rest\_Sport\_cond\_list).

check(\_,\_,[Sport\_cond|\_]):-

no(Sport\_cond),!,fail.

check(Rule\_number,History,[Sport\_cond|Rest\_Sport\_cond\_list]):-

cond(Sport\_cond,Text),

ask\_question(Sport\_cond,Text),

check(Rule\_number,History,Rest\_Sport\_cond\_list).

check(\_,\_,[]).

do\_answer(Cond\_number,1):-!,

assertz(yes(Cond\_number)).

do\_answer(Cond\_number,2):-!,

assertz(no(Cond\_number)),fail.

/\* Исключение данных из базы знаний

после завершения цикла "Распознавание-действие" \*/

erase:-retract(\_),fail.

erase.

/\* Уничтожение в базе данных всех ответов yes (да) и no (нет) \*/

clear:-retract(yes(\_)),retract(no(\_)),fail,!.

clear.

%BEGIN Task Window, e\_Create

task\_win\_eh(\_Win,e\_Create(\_),0):-!,

%BEGIN Task Window, InitControls, 21:26:22-22.12.2007, Code automatically updated!

%END Task Window, InitControls

%BEGIN Task Window, ToolbarCreate, 21:26:22-22.12.2007, Code automatically updated!

tb\_project\_toolbar\_Create(\_Win),

tb\_help\_line\_Create(\_Win),

%END Task Window, ToolbarCreate

ifdef use\_message

msg\_Create(100),

enddef

!.

%END Task Window, e\_Create

%MARK Task Window, new events

%BEGIN Task Window, id\_file

task\_win\_eh(\_Win,e\_Menu(id\_file,\_ShiftCtlAlt),0):-!,

erase,

assert\_database,

do\_consulting,!.

%END Task Window, id\_file

%BEGIN Task Window, id\_help\_contents

task\_win\_eh(\_Win,e\_Menu(id\_help\_contents,\_ShiftCtlAlt),0):-!,

vpi\_ShowHelp("logic\_es\_win.hlp"),

!.

%END Task Window, id\_help\_contents

%BEGIN Task Window, id\_help\_about

task\_win\_eh(Win,e\_Menu(id\_help\_about,\_ShiftCtlAlt),0):-!,

dlg\_about\_dialog\_Create(Win),

!.

%END Task Window, id\_help\_about

%BEGIN Task Window, id\_file\_exit

task\_win\_eh(Win,e\_Menu(id\_file\_exit,\_ShiftCtlAlt),0):-!,

win\_Destroy(Win),erase,

!.

%END Task Window, id\_file\_exit

%BEGIN Task Window, e\_Size

task\_win\_eh(\_Win,e\_Size(\_Width,\_Height),0):-!,

ifdef use\_tbar

toolbar\_Resize(\_Win),

enddef

ifdef use\_message

msg\_Resize(\_Win),

enddef

!.

%END Task Window, e\_Size

%END\_WIN Task Window

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Invoking on-line Help

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

project\_ShowHelpContext(HelpTopic):-

vpi\_ShowHelpContext("logic\_es\_win.hlp",HelpTopic).

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Main Goal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

goal

ifdef use\_mdi

vpi\_SetAttrVal(attr\_win\_mdi,b\_true),

enddef

ifdef ws\_win

ifdef use\_3dctrl

vpi\_SetAttrVal(attr\_win\_3dcontrols,b\_true),

enddef

enddef

vpi\_Init(task\_win\_Flags,task\_win\_eh,task\_win\_Menu,"logic\_es\_win",task\_win\_Title).

%BEGIN\_TLB Project toolbar, 18:45:04-21.12.2007, Code automatically updated!

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Creation of toolbar: Project toolbar

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

clauses

tb\_project\_toolbar\_Create(\_Parent):-

ifdef use\_tbar

toolbar\_create(tb\_top,0xC0C0C0,\_Parent,

[tb\_ctrl(id\_file\_new,pushb,idb\_new\_up,idb\_new\_dn,idb\_new\_up,"New;New file",1,1),

tb\_ctrl(id\_file\_open,pushb,idb\_open\_up,idb\_open\_dn,idb\_open\_up,"Open;Open file",1,1),

tb\_ctrl(id\_file\_save,pushb,idb\_save\_up,idb\_save\_dn,idb\_save\_up,"Save;File save",1,1),

separator,

tb\_ctrl(id\_edit\_undo,pushb,idb\_undo\_up,idb\_undo\_dn,idb\_undo\_up,"Undo;Undo",1,1),

tb\_ctrl(id\_edit\_redo,pushb,idb\_redo\_up,idb\_redo\_dn,idb\_redo\_up,"Redo;Redo",1,1),

separator,

tb\_ctrl(id\_edit\_cut,pushb,idb\_cut\_up,idb\_cut\_dn,idb\_cut\_up,"Cut;Cut to clipboard",1,1),

tb\_ctrl(id\_edit\_copy,pushb,idb\_copy\_up,idb\_copy\_dn,idb\_copy\_up,"Copy;Copy to clipboard",1,1),

tb\_ctrl(id\_edit\_paste,pushb,idb\_paste\_up,idb\_paste\_dn,idb\_paste\_up,"Paste;Paste from clipboard",1,1),

separator,

separator,

tb\_ctrl(id\_help\_contents,pushb,idb\_help\_up,idb\_help\_down,idb\_help\_up,"Help;Help",1,1)]),

enddef

true.

%END\_TLB Project toolbar

%BEGIN\_TLB Help line, 18:45:04-21.12.2007, Code automatically updated!

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Creation of toolbar: Help line

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

clauses

tb\_help\_line\_Create(\_Parent):-

ifdef use\_tbar

toolbar\_create(tb\_bottom,0xC0C0C0,\_Parent,

[tb\_text(idt\_help\_line,tb\_context,452,0,4,10,0x0,"")]),

enddef

true.

%END\_TLB Help line

%BEGIN\_DLG About dialog

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Creation and event handling for dialog: About dialog

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

constants

%BEGIN About dialog, CreateParms, 14:41:14-18.4.2022, Code automatically updated!

dlg\_about\_dialog\_ResID = idd\_dlg\_about

dlg\_about\_dialog\_DlgType = wd\_Modal

dlg\_about\_dialog\_Help = idh\_contents

%END About dialog, CreateParms

predicates

dlg\_about\_dialog\_eh : EHANDLER

clauses

dlg\_about\_dialog\_Create(Parent):-

win\_CreateResDialog(Parent,dlg\_about\_dialog\_DlgType,dlg\_about\_dialog\_ResID,dlg\_about\_dialog\_eh,0).

%BEGIN About dialog, idc\_ok \_CtlInfo

dlg\_about\_dialog\_eh(\_Win,e\_Control(idc\_ok,\_CtrlType,\_CtrlWin,\_CtrlInfo),0):-!,

win\_Destroy(\_Win),

!.

%END About dialog, idc\_ok \_CtlInfo

%MARK About dialog, new events

dlg\_about\_dialog\_eh(\_,\_,\_):-!,fail.

%END\_DLG About dialog