Simple Expert System Shell Running on Mobile Information Terminal Device. Data Organization

Margarita Teodosieva, Georgi Krastev

Abstract: *This paper describes the date organization of the developed light shell for expert system working on a mobile information terminal.*

Keywords: *expert system, mobile information terminal, production rules.*

1. INTRODUCTION

MIDlet application is one of the most accessible and effective ways for extending a mobile phone functionality. How it will serve, an assistant for doing business activities or entertaining device, it is just up to the user.

MIDletPascal [1. 5] is a Pascal-like programming language designed for development of mobile applications. MIDletPascal compiler translates the Pascal output code into a byte-code Java™ micro edition (J2ME). Programs written in MIDletPascal can be run on any mobile information terminals (devices, phones) which support Mobile Information Device Profile (MIDP) 1.0 and Connected Limited Device Configuration (CLDC) 1.0 platforms.

The shell created by the authors in MIDletPascal and described herein represents an expert system [2,3,4,6,7,8] running in environment of mobile information terminal device.

MIDP defines a Record Management System modeled on a basic record-oriented database.

The developed simple expert system shell uses the following data types:

- Data that has to be stored longer because of its multiple usages. It is organized in record archives stored in the flash memory of the mobile device.
- RUL data archive – it contains processed expert concepts concerning the subject domain in the following format:

  Questions to the object:
  QUESTION<OBJECT>=<QUESTION TEXT>

The question text length is up to 256 characters, each line being not more than 80 characters. The "/" character is used to mark the end of a line to a given question.

- legal object values:
LEGAL <OBJECT> = <VALUE 1>, <VALUE 2>...
For example:
legal (c1) = true, false

- production rules for inference:
IF < CONDITION > THEN < RESULT >
The rule condition and result are in the form of facts:
<OBJECT> =<VALUE>, CL(confidence level) = < VALUE >, note that facts conjunction is also allowed.

The "AND" character is used to mark a conjunction. The end of a Rule is marked with "." e.g.

```
IF
      c1 = true
      c2 = false
THEN
      damage = g1
      damage = g2.
```

- file .DEF

The record archive is used to facilitate the display of the consultation results. The archive data is organized in the following format:

<OBJECT-GOAL VALUE > =<TEXT FOR DECODING >

Decode text length is up to 256 characters.

The "/" character is used to designate that a given line belongs to it.

e.g.

g1 = worn-out crank

g2 = faulty cooling system /scale, thermostat, driving belt, radiator/

2. DATA INTERNAL REPRESENTATION

List structures are used for data internal representation. Two main list types are created – objects list and rules list.

Objects internal representation structure is shown on Fig. 1a.

Each node contains the following data:

- object name /name/
- question to the object /question/
- type of object multiple values /multivalid/
- value list top pointer /value_list/

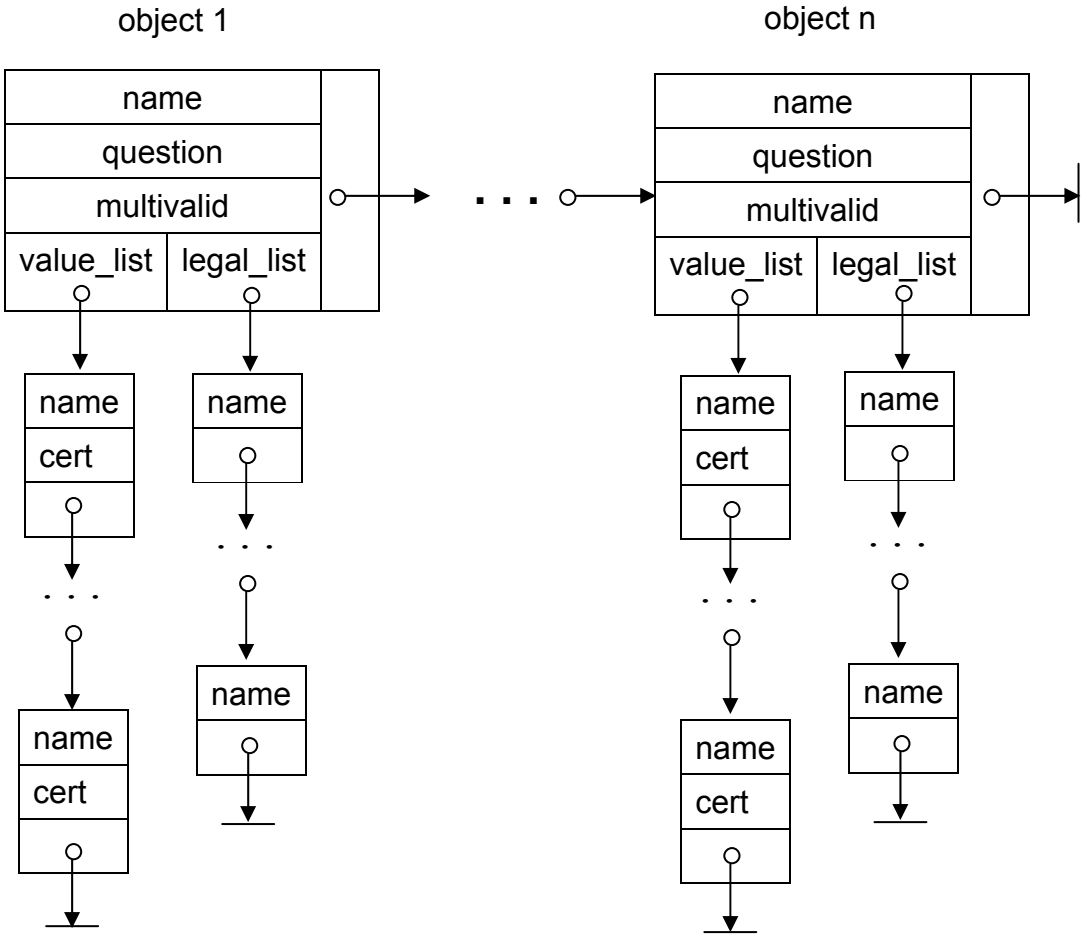

Fig.1a. Structure of objects internal representation

Each node contains data of the value /name/, its confidence level /cert/, pointer to the next object value /next/.

- Pointer to the object legal list top /legal_list/

Each node contains data of the legal value /cert/ and pointer to the next legal value /next/

- Pointer to the next node of the object list /next/

The structure of rules internal representation is given on Fig. 1b.

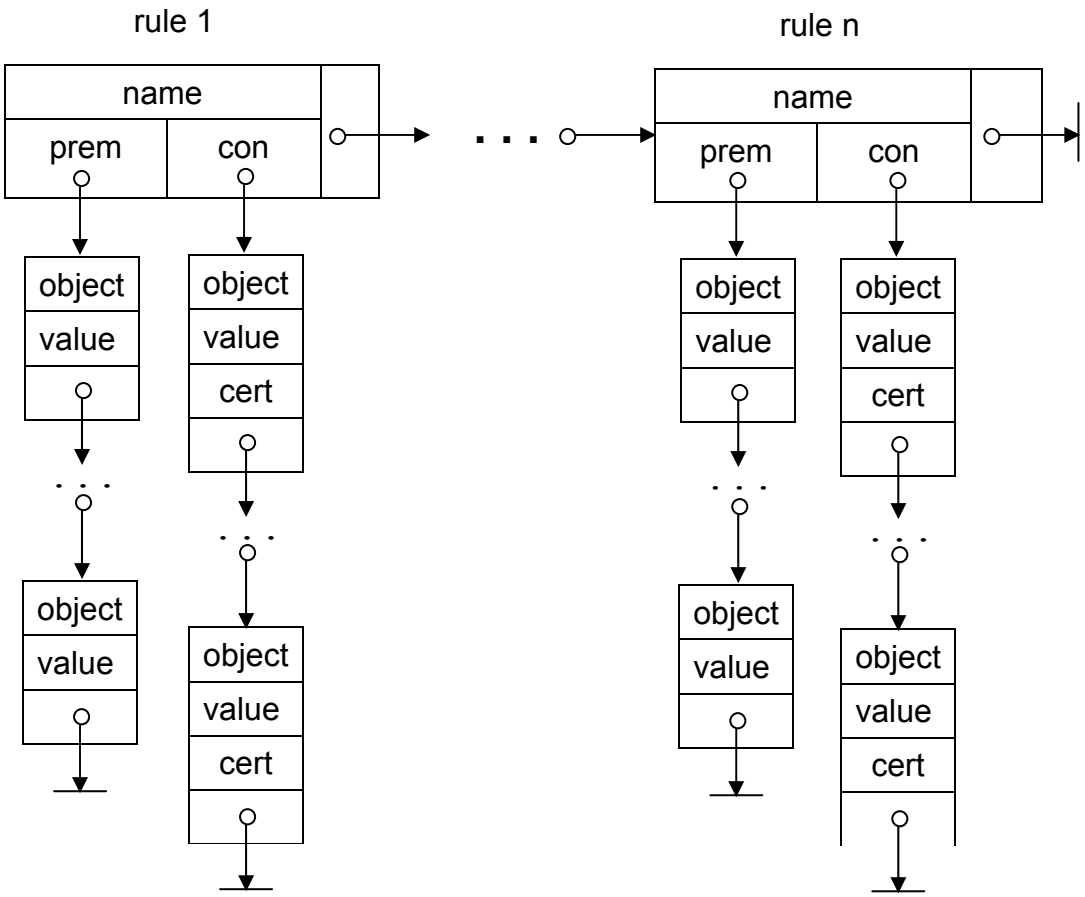

Fig.1b. Structure of rules internal representation

Each node of the rules list contains the following data:

- Rule name /name/
- Pointer to the preconditions list top /prem/

Each node of the preconditions list contains data of the object name /object/, its value /value/ and pointer to the next precondition /next/.

- *Pointer to the conclusions list top /con/*

Each node of the conclusions list contains data of the object name /object/, its value /value/, conclusion confidence level /cert/ and pointer to the next conclusion of the list /next/.

- *Pointer to the next rule /next/*

For building up the above structure the following types are declared in the program:

1 – for variables limits of string type:
WORD_STRING – variable length up to 40 characters
LINE _ STRING – line length up to 80 characters
2 – for pointers to records type given below:
VALUE_PTR = ^VALUE
Pointer to objects values list
LEGAL_PTR = ^LEGAL_VALUE
Pointer to objects legal values list
OBJECT_PTR = ^OBJECT
Pointer to objects list
PREM_PTR = ^PREM
Pointer to rule preconditions list

CONP_PTR = ^CON
Pointer to rule conclusions list

RULE_PTR = ^RULE
Pointer to rules list

3. RECORD TYPES

a/ OBJECT = RECORD
NAME : WORD_STRING;
QUESTION : STRING;
MULTIVALD : BOOLEAN;
LEGAL_LIST : LEGAL_PTR;
VALUE_LIST : VALUE_PTR;
NEXT;
The record fields have the following values:
NAME – object name
QUESTION – question text to the object
MULTIVALID – flag of multivalue
TRUE – the object is multivalued
FALSE - the object is not multivalued
LEGAL_LIST – pointer to the legal values top
VALUE_LIST – pointer to the object values list

b/ VALUE = RECORD
NAME : WORD_STRING;
CERT : INTEGER;
NEXT : VALUE_PTR;
The record fields have the following values:
NAME – object values;
CERT – value confidence level

c/ LEGAL_VALUE = RECORD
NAME : WORD_STRING;
NEXT : LEGAL_ PTR;
The field value is the following:
NAME – for the extended value

d/ PREM = RECORD
OBJECT : WORD_STRING;
VALUE : WORD_STRING;
NEXT : PREM_PTR;
The record fields have the following values:
OBJECT – precondition object name
VALUE – object value

e/ CON = RECORD
OBJECT : WORD_STRING;
VALUE : WORD_STRING;
CERT : INTEGER;
NEXT : PREM_PTR;
The record fields have the following values:
OBJECT – conclusion object name
VALUE – object value
CERT – confidence level value for the conclusion

f/ RULE = RECORD
NAME : WORD_STRING;
PREM : PREM_PTR;

```
    CON          : CON_PTR;
    NEXT         : RULE_PTR;
```

The record fields have the following values:
NAME – rule name
PREM – pointer to preconditions list top
CON – pointer to the conclusions list top

g/ DEF = RECORD

```
    NAME_DEF    : STRING;
    VAL_DEF          : WORD_STRING;
    CF_DEF           : INTEGER;
```

The record fields have the following values:
NAME_DEF – object-goal value
VAL_DEF – text for decoding
CF_DEF – confidence level of the object-goal value

4. CONSTANTS

WORD_MAX = 40 – number of characters in the character line
COLON = " : "
PERIOD = " = "
COMMA = " , "
SPACE = " "
EQUALS = " ="
SLASH = "/"
DEFINITE = 100

5. DATA USED DURING PROGRAM EXECUTION

a/ variables of INTEGER type:
= variables used for storage of:
st_left, st_right, com_place – position of a character found in a character line
len – length of a character line
cf, cf1, cf2, prem_cert, lowest –confidence level read
pick - for the current number of object legal value
= variables used for counters:
num_vals – for object legal values
n – for object-goal values
= variables used for cycles organization:
i, j, x

b/ variables of STRING type:
line – storage of questions and object-goals values
type WORD_STRING
name_file, name_deffile – names of files
s_object, s_value, s_word, dummy, word – names of resp. objects, values, meanings
rule_name – rule name
type LINE_STRING
s_line – for storage of a character line

c/ variables of CHAR type
c – for storage of a current character of a character line
ch – for storage of a read character of the keyboard

d/ variables of POINTER type
top_fact – pointer to the objects list top
top_rule – pointer to the rules list top

last_tru – pointer to the last object searched in the objects list
curr_object – pointer to a current object
curr_value - pointer to a current object value
curr_rule - pointer to a current rule
curr_prem, curr_con - pointer to the precondition and conclusion of a concrete rule

e/ variables of BOOLEAN type
explain – flag for conclusion reasoning
true - consultation with conclusion reasoning
false - consultation without conclusion reasoning
cf_on - flag for result display
true - with confidence level
false - without confidence level
done - flag for cycle ending when a value is not found
suppress-flag for differentiation of the line processing, having question to the object
found - flag for found rule, containing object with a settling value
solved – flag for non-found object-goal value
bod - flag for fixing a new goal
e: - variable of ARRAY type
def - array containing data of the object-goal values

f/ variable of TEXT type
def_file, rules – variables for operating with DEF and RUL records archives

6. CONCLUSIONS

1. The built-in possibility for setting up a multivalue of an object allows the processing of unspecified and indefinite data.

2. The built-in possibilities for conclusions reasoning make the expert system transparent and comprehensive to the user and allow its usage for diagnostics aims.

3. The reasoning capacities of the ES can be further developed with respect to providing more information to the user concerning the conclusion tracking, goals tracing, objects values displaying, etc.

REFERENCES:

[1] G. Krastev, M. Teodosieva. Programming of mobile information terminals with MIDLetPascal. PH University of Ruse, ISBN 954-9906-68-X, 2008.
[2] Jackson P., Introduction to Expert Systems. Moscow: Вильямс, 2005, -623 p.
[3] Aikins J.S. Prototypical knowledge for expert systems. Artificial Intelligence, 2003, 210 p.
[4] Buchnan B.G. Constructing an expert system. Reading MA: Addison-Wesley, 485 p.
[5] http://www.midletpascal.com/
[6] http://www.expertsystem.net/
[7] http://www.pcai.com/web/ai_info/
[8] http://eu.wiley.com/

ABOUT THE AUTHORS

Assoc.Prof. Margarita Teodosieva, PhD, Department of Informatics and information technologies, University of Rousse, Ph: +359 888 490, E-mail: mst@ami.ru.acad.bg.

Assoc.Prof. Georgi Krastev, PhD, Department of Computer Systems, University of Rousse, Ph: +359 888 672, E-mail: gkrastev@ecs.ru.acad.bg.