On Specialized Programming Language for Flexible Information Systems

Aleksandar Popovic, Zoran Budimac, and Mirjana Ivanovic

It is often the case that processing of some data classes = needs parameters. However, there are no easy solutions if processing = parameters must be expressed by algorithm (or program) and not by = "ordinary" data. In this paper a specialized programming language = (called PLES) that solves the problem is described. Programs of this = language are stored into the database together with the rest of the = data. Programs can be changed and executed without any support from = development team or development environment of the information system. = By using programs of this specialized programming language, information = systems can be more flexible and significantly easier to maintain. = Half-finished software components can be easily built as well.

Programming language PLES contains:

* several built-in simple data types (fixed point numbers, strings, = booleans, characters, etc.)
* two built-in structured data types (arrays and records)
* subroutines (procedures and functions)
* several control structures (if-then-else, while loop, for)
* a possibility to access "external" data (i.e., a data from the PLES = program environment: fields of a screen form or database tables). This = access is compatible with the access to ordinary PLES program variables.

Depending to which external data PLES program can access, PLES is = defined in two variants: PLES-Client and PLES-Server. PLES-Client = programs can access only screen form fields and can be executed at the = client side of an information system. PLES-Server programs can access = only database table fields and can be executed at the server side of an = information system.

PLES is implemented as two components:

* a compiler into a machine language of an abstract machine and
* an interpreter of machine language of that abstract machine

The choose solution is appropriate, because of the following reasons:

* It is easily portable.=20
* Abstract machine code can be easily and quickly transmitted through a=20
* Execution of abstract machine code is more efficient that a direct = interpretation of PLES programs.