INTEGRATED ENVIRONMENT OF SYSTEMS AUTOMATED ENGINEERING.

Autores
Arturo Carlos SERVETTO, Ramón GARCÍA MARTÍNEZ, Gregorio PERICHINSKY
aserve/rgm/gperi@mara.fi.uba.ar
Tel. (011) 4342-9184 / 4343-0891, Int. 142
Laboratorio de Bases de Datos y Sistemas Operativos
Facultad de Ingeniería
Universidad de Buenos Aires

KEYWORDS: static models, dynamic models, integrated environment, conceptual pattern.

ABSTRACT.

The Standardization of a development process for systems of medium complexity, entirely supported by an integrated development environment, could mean a double contribution to the industry of systems development. On the one hand it would contribute to the reduction of costs by shortening times of development and facilitating the maintenance of the systems. On the other hand, by using conceptual models of a high level of abstraction, final users would be able to visualize and follow the progress of systems development and to commit themselves in the process actively, guaranteeing their success.

The purpose of this series of papers is to identify, design, develop and integrate the components of an integrated environment for a system automated development, starting from high-level-abstraction formal specifications. It is intended to achieve a generation of systems starting from only two models: the static or data structure model, and the dynamic or functional model. The former is based on an adaptation of the conceptual pattern of entities and relationships, and the latter on the formal specification of operations in objects relational algebra and on the finite automaton theory. The maintenance of the systems generated by the tool would be made by operating directly on the static and dynamic models, with no need for either re-coding or making reverse engineering.

The strength of the proposal is based on integration of the Software Engineering and the analysis and metrics of quality. Storage protocols of data definitions and storage protocols of interface definitions corresponding to a graph, through the state transitions and their gradation so that the user could visualize the development of the life cycle[6][32][33][47]. Any artifact or document can be measured and estimated if it is produced during the software life cycle.

This study is an approximation to a process of activities supported by an effective help tool. Abstraction and visualization capacities are provided to the artifacts with the system design and analysis (CASE). Such artifacts under controlled conditions trace the software life cycle taking information of the structure (framework) of the application; with Quality, Reliability and Metrics.

PROCESS ORIENTED.

Similarly to what happened with the imperative programming and the structured systems development, at the beginning the Object Oriented (O-O) paradigm was applied exclusively to the programming. The following phase, more or less in the 80s, was characterized by the interest explosion in the interfaces of the user. The most evident effect is reflected in the current abundance of libraries of classes for the development of interfaces[24][36][38].

The current phase of the history of the O-O is characterized by the emphasis which has been moved from programming to analysis and design, and because there is conscience of the problem of the open systems and of the need for standards. There exists an important trend toward the incorporation of methods guided to objects, in the systems of database management as well as in the structured methods of existing management and the CASE tools that give support to systems of database and structured methods[2]. The proposal of this work consists in the specification of a process and behavior model of systems for a CASE tool that, based on the analysis by scenarios and in the classification of objects in a system in objects of application and of interface, synthesizes the
functionality and the behavior of a system in a finite states machine by scenery [1][9][11][17][18][25][26][34][35][37][39][45][46][53].
The states in the model represent objects of interface and the transitions represent messages of activation to other objects of interface or well messages to classes of application that give as a result a set of objects that is associated with an object interface. At the same time each object interface, associated with an empty set, unitary or vast of objects of application, it can have some of these active objects or in area (inclusive null or undefined object).

Storage protocols of data definitions and the definitions grammar corresponding to a graph can be simple (defined attribute, aggregate entity, aggregate attribute, identifier and hierarchy) or composites (attributes, entities and relations). Another protocol are the storage protocols of interface definitions and the definitions grammar corresponding to a graph can be simple (report, attribute value, computed value, conjunction term and code) or composites (interfaces, forms, transitions, relations between forms and a disjunction term).

An adjustment is used for the formal specification of the messages to classes of application of relational algebra, whose operations determine instance subsets of classes of application or joint of instances of temporary classes, created as a result of consulting. The model results apt for the automation of the office-supply systems development or from traditional management and reduces the behavioral specification and functionality of a system to an only graph by all scenarios, synthesizing the scenarios or cases of use, the interaction among objects (sequence graphs and of collaboration) and the state sequences, proposed by the UML (Unified Modeling Language) of Booch, Rumbaugh and Jacobson.

REFERENCES