

SALVO—a fourth-generation language for personal computers

by MARVIN ELDER
Software Automation, Inc.
Dallas, Texas

ABSTRACT

Personal computer users are generally nontechnical people. Fourth-generation products can be of great assistance to these users, especially to those who have no access to database administrators or other computer professionals.

SALVO is a fourth-generation language for personal computers. This product was developed over a three-year period. Since the first working prototype (August 1982), this product has evolved into areas of artificial intelligence (AI), particularly natural-language processing and expert systems. The addition of AI functions to a fourth-generation language represents a departure from most fourth-generation products written for mainframe computers (except INTELLECT).

The synergism of this new area of AI research, coupled with relational database management, has proved to be extremely beneficial in assisting end users: they can extract information and generate applications in a much more nonprocedural manner than with more conventional fourth-generation approaches.

INTRODUCTION

SALVO is a fourth-generation language that runs on a wide variety of personal computers, both in a standalone mode and in an office automation environment.

The product combines several advanced software technologies, most notably relational database management and artificial intelligence. Its principal target is the end user who wants to manage and retrieve information without attending to the underlying technical details of data processing. At the same time, it will appeal to professional programmers who want to improve their productivity. Contrary to many “friendly” packages, SALVO’s ease of use has not been purchased at the expense of power and functionality.

To achieve its intended purpose, SALVO’s developers have automated most of the how-to functions, allowing the end user to concentrate on what information is required. This design philosophy has resulted in a product that differs in many respects from fourth-generation products designed for the mainframe environment, where database administrators and other data processing professionals can provide technical support for corporate end users.

The technical features of SALVO are arranged and described in the following categories:

1. Relational database management features
2. Application generator features
3. Artificial intelligence features
 - a. Natural language processing
 - b. Expert systems
4. Graphic automated information management

Before each set of features is described in detail, an overall perspective of SALVO will be presented by briefly stating the designers’ working notion concerning the difference between *information* and *data*.

Information is entered into a computer in discrete pieces of datum elements. These data are best stored and managed by a relational database management system (for reasons explained below). Processing these pieces of data requires a “procedural language” (even fourth-generation languages must have procedural capabilities). Retrieving information from a system, on the other hand, is possible through a non-procedural language.

Most end users who want to get information from a computer do not want to perform any detailed data processing functions. A system that automates all of the details of data processing would understand natural language at the level of a human being. Of course, there are some jobs that even human beings have to perform in a procedural, algorithmic fashion (e.g., compute the payroll).

A major hypothesis according to which SALVO is designed is this: If data are organized and managed by a relational database management system—one that is truly relational, according to Codd’s definition¹—then an expert system (having the expertise of a database administrator and programmer) can automate many of the detailed how-to functions involved in data processing and can retrieve and manage information for the user. SALVO’s designers call this “software automation.”

This approach of combining relational database theory and artificial intelligence theory actually surpasses the notion of fourth-generation languages and approaches the fifth-generation ideas now being formulated and researched in some areas.

The first two categories of features in SALVO discussed below—relational database management and application generation—concern themselves with the data management and data processing functions (i.e., the procedural aspects) required of a fourth-generation product.

The third and fourth categories, artificial intelligence and user interface features, are concerned with the nonprocedural information management functions—of particular importance to personal computer users.

RELATIONAL DATABASE MANAGEMENT FEATURES

There are many so-called relational database management software packages that run on personal computers. Most of these are not true relational systems. According to E. F. Codd’s definition, a true relational system has the following components:

1. A relational data structure (i.e., flat files, relations, tables)
2. A collection of relational algebraic functions, or rules of inference
3. A collection of relational integrity rules

In light of the criteria stated above, the majority of “relational” software packages for personal computers can be broadly characterized by noting that many of them do keep data in relations (although normalization is usually not enforced); some of them provide most of the relational algebraic operators; and few (perhaps none) of them enforce the relational integrity rules.

An important objective of the relational data model, according to Codd, was to make the model structurally simple so that users and programmers could communicate with one an-

other about the database (Codd calls this the "communicability objective").

Without going into detailed discussion about the merits (and drawbacks) of the relational model, the most important reasons for using it in SALVO are the following:

1. If data are kept in normalized relational form, with their integrity enforced by the relational integrity checks, then a set of inference rules (the relational algebra functions) can be installed as an expert system that can find and manipulate related pieces of data in the database.
2. Therefore users can more easily understand the structure and relationship of their database and can formulate requests for information that may be accurately inferred by SALVO'S expert system and processed through its concept of software automation.

Despite the many advantages of the relational model over other database models, critics often point out that relational database systems are inherently slow in execution. SALVO has three aspects of relational database management designed to overcome certain inherent disadvantages of the relational data model.

Virtual Join

The relational algebraic operation called join compares the values of a data field common to two tables (files) and produces a third *result table*, which may be much larger than either of the two tables being joined. In some cases a result table may be too large to fit on a floppy disk on a personal computer. To avoid this common problem, SALVO employs a Virtual Join, in which the result of a join is not a physical table but the output device itself (CRT screen or printer).

In conventional systems, join operations over several tables or files must be performed sequentially. SALVO's virtual join operation can simultaneously join up to 16 tables in a single pass, resulting in dramatic timesaving compared with other systems.

Automatic ISAM Indexing

Before attempting a virtual join operation, SALVO automatically builds ISAM indexes (if they are not already built) that speed up the execution of the join operation.

Automatic Normalization

In a large company using a mainframe relational system, a highly paid database administrator usually sets up and maintains the database structure. This individual uses and enforces an integral part of database design called normalization. Relational databases should be maintained in third normal form (or higher) to avoid storage anomalies that can result in lost or redundant data in conventional, unnormalized databases.

SALVO employs a sophisticated automatic normalization concept, since the average user of a personal computer does not have a database administrator to consult. This unique feature hides the complexities of normalization from the user;

in fact, the user is not even aware of this operation at all.

To summarize its relational database features, SALVO is thought to be one of the first true relational database management systems running on personal computers. This rigid enforcement of the relational model provides a basis for SALVO's automatic navigation, expert system, and other advanced information management concepts. SALVO also employs sophisticated database methods not generally found on mainframe computers, much less on personal computers with only 64 Kbytes of memory!

Application Generator Features

A fourth-generation language that is entirely nonprocedural will allow users to retrieve information without detailed programming but will be limited to queries only. To develop applications that involve any logical decisions and/or the processing of data (e.g., storing, adding), a language must have procedural aspects. Since personal computer users want to be able to do many of their own computer applications without the assistance of professional programmers, a simple but effective procedural capability is a requirement of a fourth-generation language.

The difference between a procedural fourth-generation language and the third-generation languages (such as COBOL and BASIC) is the number of procedural instructions necessary to write an application.

James Martin, in his book *Application Development Without Programmers*,² defined a fourth-generation procedural language as one that requires fewer than one-tenth as many instructions as present-generation languages.

SALVO does much better than Martin's minimum requirements. Many benchmark programs have been written that compare SALVO with other languages. For the type of applications needed for personal computers, SALVO has been often found to be 30 to 40 times more powerful than BASIC or COBOL. Even when compared with "database languages" that run on personal computers, SALVO is often found to be 3 to 10 times as powerful.

Much of SALVO's power as an application generator is derived from its automatic navigation feature. Whereas other languages require many lines of instruction just to open a file, read a record, and test some data element to match up related records with a "foreign" file, SALVO does this type of job automatically. Especially when several files are to be related in a report, SALVO's automatic navigation (combined with its virtual join feature) can save hundreds of lines of code compared with other languages.

Professional programmers are of course interested in fourth-generation procedural languages for the same reason—fewer lines of code do the same job. On a personal computer, a fairly simple application written in COBOL (say a payroll) could easily cost more to program than the computer itself! SALVO has very powerful commands for the professional programmer. Many users will never use some of these advanced features. Other reasons than simply fewer lines of code are also important to software developers: no need for specification documents and less need for documentation, for example.

ARTIFICIAL INTELLIGENCE FEATURES

The field of artificial intelligence (AI) comprises many diverse areas. Two particular areas of AI are important to the development of user languages: natural language processing and expert systems.

SALVO contains elements of both of these AI concepts; however, SALVO does not purport to be a sophisticated AI product. In fact, AI concepts are used in SALVO only to the extent that the user interface portions of the package are made simpler, as will be explained below.

Natural Language Processing

Computer programming languages are notoriously insistent upon a rigid syntax: The commands in the language have to be stated in a precise order of words. Human language is much richer and diverse in the way that an idea may be expressed.

SALVO has a Request facility, which allows users to ask for information (i.e., make a query) in natural language. This facility is not intended to allow purely “conversational” queries—as if one were talking to another human being. The major benefit of the Request facility is to state a query in a straightforward way, using declarative natural language commands, without having to obey the syntax rules of a computer language.

Using Request, a user can state a query that can be run immediately and never see the underlying procedural language, which is actually translated from the natural language request. Alternatively, the user can view (and then modify) the SALVO procedural template program, which the natural language processor built from his or her request.

SALVO’s natural language processor employs a frame-and-slot approach to decompose the user’s request into a format appropriate for an internal expert system (explained in the next section).

This AI method allows the user a considerable amount of freedom in the way his or her request is stated. As an example, if the user does not include a verb in a command, a default verb (list) is supplied automatically. In SALVO the following three requests are identical: “List the accounts for salesman Smith,” “For salesman Smith, list his accounts,” and “Salesman Smith accounts.” These examples illustrate the degree of syntactical freedom allowed in SALVO’s natural language processor.

Expert Systems

An expert system emulates the decision making and the knowledge of a human expert in a particular field. Two major components of an expert system are (1) a means of making inferences based on a set of rules (formal and/or intuitive) that the expert uses to make decisions and (2) a knowledge base that models the knowledge accumulated by the expert pertaining to his or her field of expertise.

SALVO contains an expert system with the following components:

1. A set of inference rules that a database administrator would use
2. A knowledge base consisting of the data dictionary of the user’s particular database

SALVO’s expert system is utilized in three different areas of the software:

1. Translating a user request from natural language to a SALVO procedural template program
2. Assisting a user or programmer in setting up his or her database (i.e., automatic normalization)
3. Compiling a SALVO procedural program into executable form—specifically, deciding internally how to handle most of the details of conventional programming.

GRAPHIC “AUTOMATIC INFORMATION MANAGEMENT”

The research and development of SALVO, over a three-year period, evolved in ways that were unanticipated at the outset. One discovery that evolved out of the project is a new working model of information processing, not previously published (to our knowledge) in the literature.

This new working model of information processing is roughly stated as follows: Given a relational database, given a perspective of looking at that database from a particular user’s viewpoint, and given an expert system that “knows” the rules of relational database navigation, it is possible to determine what information can be derived for just this particular view.

SALVO’s developers have named this new approach “automatic information management” and have provided a graphics user interface to implement this approach.

This facility in SALVO is accessed through a function called “view.” First the user selects a particular file that serves as the focus point of his or her intended request for information. A graphic representation of the user’s view of the primary file and its related files is then displayed. A natural language request is started for the user, first of all to simply list the primary file selected. The user can then include information from other related files in the request by simply selecting from the graphic display of the files.

As each related file is selected, the user’s request is built automatically in natural language, just as if the user had typed the query through the request function.

The view function of SALVO allows people with no programming experience, and with no desire to see or manipulate even a fourth-generation language, to request and generate information from their personal computers.

OPERATING ENVIRONMENTS

SALVO incorporates several state-of-the-art features into a software package that runs in a 64-Kbyte environment on a personal computer. Written in FORTH, this new fourth-generation language is transportable to many popular types of microcomputers and/or operating systems.

The first version of SALVO is designed for a single user personal computer system. SALVO-II, a version available in the first quarter of 1984, has features desirable in distributed database processing environments (i.e., the office automation environment).

SUMMARY

Personal computers need a particular type of fourth-generation language—one that end users can use without any help from programmers, database administrators, or other computer professionals.

The combination of two advanced software technologies,

relational database management and artificial intelligence, provides a synergistic approach to information management that is perhaps more powerful than either of these technologies by itself has been able to achieve. SALVO may be the first of this type of fourth-generation product to run on a personal computer.

REFERENCES

1. Codd, E. F. "Relational Database: A Practical Foundation for Productivity." *Communications of the ACM*, 25 (1982), p. 111.
2. Martin, J. *Application Development Without Programmers*. Englewood Cliffs, N.J.: Prentice-Hall, 1982.