DATA ABSTRACTIONS FOR DATA BASES*

Michael Hammer
MIT Laboratory for Computer Science
Cambridge, Mass. 02139

The concept of abstract data types has emerged from programmng language research as a device to encourage and facilitate structured and modular programming [1]. It separates the abstraction of a data object from its implementation. The user of an abstract data type only concerns himself with the behavioral semantics of the type: what meaningful operations can be applied to objects of the type; the internal representation and structure of the type is unknown to him. In this way, irrelevent detail is suppressed and meaningfulness of programs enhanced.

What application does this concept have to the area of data bases? There are some obvious similarities between the principle of abstraction and the concept of data independence of data bases [2]. Data independence means that application programs that utilize a data base do not need to know either the physical or the structural organization of the data base, but can relate to it purely on a logical plane. As with data abstractions, the goal is clarity and modifiability of programs, resulting from concentration on semantics rather than representation.

The relational model of data, in particular, attempts to achieve a high degree of data independence, by providing the data base user with a simple, uniform view of the data's structure [3]. The concept of data independence is also manifested in the recommendations of the ANSI/SPARC Standardization Committee, which has proposed internal, external, and conceptual schema levels for data management systems [4]. Each of these views of the data base represents a different "level of abstraction" and is used for different purposes. The internal schema describes the organizational structure and storage strategy of the data. The external schema specifies the view of the data base as seen by an application programmer; different application programmers, using the data base in different ways, might consequently utilize different external schema descriptions. But each of them must be consistent with the conceptual schema, which specifies the enterprise's view of the structures being modelled by the data base; in some sense, this level represents the most fundamental semantics of the data. Research is being conducted to determine the best mechanisms to describe the structure of a data base at these levels of semantic abstraction [5].

But there is an important difference between the concept of data abstraction and the higher-level data models being proposed for the conceptual and external schema levels; the difference is that between behavior and representation. Current approaches to higher level data models attempt to provide conceptual data structures, rather than physical ones, with which to model the relevant application system; but these models are nonetheless representational, and describe the structure of the data at some level. However, an abstract data type does not present its users with any view of its structure, high or low level; its semantics are entirely behavioral.

The data independence of data bases could be enhanced by making use of the operational view of data semantics which is inherent in abstract data types. Some of the advantages are obvious. The simplicity and reliability of application programs would be increased by limiting their access to the data base to a fixed number of semantically meaningful operations. For example, a personnel file might only be accesible by means of problem-oriented operations as "hire", "fire", "give raise", etc. This would forestall an application program from attempting (either deliberately or accidentally) to add a salary field to a date-of-birth, or perform some other meaningless activity. In addition, an operational interface to the data base immunizes the user to changes, not only in the data's physical structure, but in its logical organization as well. For example, a redistribution of field-types among the record-types would be reflected only in the implementation of the abstract operations, not in their use. More generally, viewing the data base in behavioral terms provides a useful structure with which to address certain complex problems. For example controlling the (semantic) integrity of a data base can be greatly facilitated by restricting the range of possible

*This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Office of Naval Research under contract number N00014-75-0661

transactions with the data base to those which are meaningful in its own terms [6]. The issue of access control should also yield to attack from this direction.

However, incorporating the operational model of data into the data base environment is a difficult research problem. The principal reason for this derives from the difference between the programming environment for which the abstract data type concept was originally formulated, and that of a shared integrated data base. In the former situation, a programmer (or programmer team) is attempting to structure a large program in a disciplined modular fashion. He follows the principle of "abstraction first, implementation later" and initially determines the behavioral semantics of his data objects. These operations completely characterize the data objects, and their implementation can be encapsulated. On the other hand, the definition of a shared data base is principally determined by the data which it contains, rather than by one particular set of operations which may be applied to it. Many important uses of a data base do not develop until a long time after its creation. Different coexisting applications may utilize radically different sets of operations. As applications come and go, the abstract view of the data base may undergo major changes, with only the data remaining constant.

In order to utilize data abstractions in data base description, several complex issues will need resolution.

1. Some interactions with a data base are not meaningful as abstract operations in the world that the data base models, but are purely related to the data base itself. For example, updating the data base to correct some faulty value in it (such as changing a date of birth) may not be the representation of a meaningful semantic activity.

2. The way in which an application programmer views and uses a data base frequently changes over time. It is thus inappropriate to provide a fixed set of operations in a programmer's external schema. A mechanism must be found to allow behavioral semantic models to evolve over time, without making old application programs inconsistent.

3. Different application programmers may use different behavioral specifications, each defining a different abstraction, but all using the same representation (internal schema); furthermore they all must be consistent with the conceptual schema. It will be necessary to isolate the different abstractions, so that they do not interfere with one another, while assuring that they all cooperate in interfacing to the other schema levels.

4. A new technique for behavioral specification will be required in order to incorporate data abstractions on the conceptual schema level. No single set of operations can possibly encompass all potential uses of the data base. Since the conceputal schema represents the broadest and most general view of the data base's semantics, a behavioral specification at this level can only constrain or similarly characterize the class of meaningful operations on the data. The operations defined in each external schema would then be instantiations of the very general templates provided in the conceptual schema. This concept could also be applicable to the programming language environment, providing for partial specifications of a type, against which any particular behavioral definition would be verified.

REFERENCES

1. Liskov, B., and S. Zilles, "Programming with Abstract Data Types", in Proceedings of Symposium on Vey High Level Languages, SIGPLAN Notices 9, 4 (April 1974).
2. Martin,J., Computer Data Base Organization, Prentice-Hal, 1975.
3. Codd, E.F., "A Relational Model of Data for Large Shared Data Bases", CACM 13, 6 (June 1970).
4. Steel, T.B., "Data Base Standardization - A Status Report", Proceedings of the 1975 ACM SIGMOD International Conference on Management of Data, May 1975.
5. Proceedings of the IFIP Working Conference on Modelling in Data Base Management Systems, January 1976.
6. Hammer, M. and D.J. McLeod, "Semantic Integrity in a Relational Data Base System", in Proceedings of the International Conference on Very Large Data Bases, September 1975.

Author's address: Prof. Michael Hammer, MIT Laboratory for Compute Science, 545 Technology Square, Cambridge, Massachusetts 02139