The Iris Database System

Bill Kent, Peter Lyngbaek, Samir Mathur and Kevin Wilkinson

Hewlett-Packard Laboratories
Palo Alto, California

Iris is an object-oriented database management system being developed at Hewlett-Packard Laboratories [1], [3] This videotape provides an overview of the Iris data model and a summary of our experiences in converting a computer-integrated manufacturing application to Iris An abstract of the videotape follows

Iris is intended to meet the needs of new and emerging database applications such as office and engineering information systems, knowledge-based systems, manufacturing applications, and hardware and software design These applications require a rich set of capabilities that are not supported by the current generation (i e , relational) DBMSs

The Iris data model is an object and function model It provides three basic constructs *objects*, *types* and *functions* As with other object systems, Iris objects have a unique identifier and can only be accessed and manipulated through functions Objects are classified by type Objects that belong to the same type share common functions Types are organized into a hierarchy with inherited functions In Iris, functions are used to model properties of objects, relationships among objects and operations on objects Thus, the behavior of an Iris object is completely specified through its participation in functions

Iris provides good separation among its three basic notions This simplifies the data model making it easier to learn and easier to implement since there are fewer constructs than other object models In addition, it facilitates Iris support for the following desirable features **Schema evolution:** new types and functions may be added at any time **Object evolution:** Iris objects may have multiple types and may acquire and lose types dynamically Object participation in functions may be required or optional (e g , everyone has birthdate but not everyone has a phone number) **Data independence:** the implementation of a function is defined separately from its interface Thus, the implementation of a function may change without affecting applications that use it **Functional extensibility:** an Iris function may be implemented as a stored table, computed as an Iris expression, or computed as a subroutine in a general-purpose programming language Thus, any computation can be expressed as an Iris function **Schema and data uniformity:** the metadata is modeled and manipulated using the primitives of the data model Also, system functions (create type, delete object, etc) are invoked in the same manner as user functions Thus, users need learn only one interface **Set processing:** Iris supports set-at-a-time processing for efficient retrieval and update of collections of objects

To evaluate the usefulness of the Iris prototype, a project was undertaken to convert a large relational application to Iris [2] The relational system contained nearly 200 relations and 2500 attributes When transcribed to Iris, the schema size was reduced by over a third There are two reasons for this large reduction First, in the relational schema, many attributes were simply foreign keys required for joins In the Iris schema, function inheritance through the type hierarchy eliminates the need for many of these foreign keys A second reason for the schema reduction was that compound keys were replaced by object references This permitted several attributes in a relation to be replaced by a single identifier

It was noted that application programs were easier to read and develop using the Iris schema The Iris OSQL (Object SQL) language was a fairly natural interface for users familiar with SQL The use of function composition and function inheritance hid a large number of joins that, in the relational system, must be expressed by comparing keys The function-orientation of Iris encouraged *code sharing* in that deriving and sharing new functions was simplified

Finally, since there are few tools and methodologies for using object-oriented database management systems, the ability of the Iris schema to easily evolve was valuable in iteratively refining the Iris schema Also, the Iris Graphical Editor was a useful tool in graphically displaying the schema and browsing function definitions and instances

References

[1] D H Fishman et al Overview of the Iris DBMS In W Kim, F H Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications* ACM Press, New York, N Y , 1989

[2] M Ketabchi, S Mathur, T Risch, J Chen Comparative Analysis of RDBMS and ODBMS Case Study In *Proceedings of Compcon IEEE Computer Society International Conference*, San Francisco, California, February, 1990

[3] K Wilkinson, P Lyngbaek, W Hasan The Iris Architecture and Implementation *IEEE Transactions on Knowledge and Database Engineering*, 2(1), March 1990