8. Status and conclusions

8.1 Summary

This document has described the Model level of the Cypress Database Management System, including the data model, the background and motivation for choosing that model, the client interface to the system, its implementation, and the database environment and applications for which it was implemented. The guiding principles in the design and implementation have been simplicity and utility for the set of applications we envision.

The result is a model that includes features of the relational model and distillations of desirable features of more recent semantic models. The model includes the concept of entities with unique names, a hierarchy of types of entities, types and uniqueness constraints on relation attributes, relational views, and a logical segmenting mechanism that can be used to facilitate physical distribution and independence of databases. This report discusses a number of issues in the implementation of these features, which are not present in any existing database system to the author's knowledge. The Cypress data model alleviates the problems motivating its development, reducing the quantity of data modelling mechanism built anew for each application, and simplifying the sharing of databases between applications. The Cypress model has also enabled the development of general-purpose tools formerly impractical due to the lack of type information and integrity checking in the Cedar database system.

8.2 Some results

Some overall statistics on the performance of the initial implementation may be helpful here. We developed two benchmark programs, one write-intensive and one read-intensive, to examine performance. Average times for the most common operations are roughly:

1 ms: GetF
1 ms: NameOf
0.5 ms: NextEntity
0.5 ms: NextRelship
5 ms: DomainSubset
7 ms: RelationSubset
10 ms: SetF
10 ms: ChangeName

These times are approximately in order of decreasing frequency of calls by the benchmarks, and

include overhead at all levels of the database and file systems. The times were taken on the Xerox Dorado, a personal super-computer with a micro-cycle time of about 60 ns. Note that since most of the Cypress operations are disk-limited, the times increase only somewhat for slower processors. The times shown above vary widely with a particular application's data schema and access patterns, so these numbers should be regarded as very rough averages. Some effects of particular optimizations and schema changes were enumerated in Section 6.

A significant result of our work is that the type checking required by the data model is *not* a large overhead in the Cypress implementation. Because we do not compile database accesses, we must check in the implementation of every operation, e.g. SetF, that the arguments passed are of the proper and coordinated types. On a SetF, for example, we must check that: (1) the arguments are a relationship, attribute, and value, respectively; (2) the attribute is of the same relation as the relationship; (3) the value is of the same type as the attribute; and (4) that a key value constraint would not be violated by the new value. The caching of information about attributes improves the performance of the first three of these considerably. Without this caching, the GetF operation takes approximately 8 times as long.

A closer analysis of the time spent in a typical read operation, e.g. GetF, is enlightening. For our benchmark programs, the time breakdown was roughly as follows:

10% model level consistency checking and access path selection
20% storage level operation: actual read or update of data
50% waiting for disk operation (cache miss)
20% other overhead (page faults, garbage collection)

Again, these proportions can vary widely with the particular application.

8.3 Status and plans

The first implementation of the Model level was completed in December of 1981, and was exercised and debugged through 1982. Approximately six man-months went into its development. This implementation includes essentially all data model features except views and augments. Views have been deferred to the development of the Query level.

Plans for the near future are to concentrate on the development of more applications, continuing the work sketched in Section 7.

Acknowledgements

Nori Suzuki and Mark Brown participated in the design and implementation of the original Cedar Database System. Peter Deutsch and Jerry Popek assisted in the design phase. Eric Bier assisted in the initial implementation of the Model level, as well as providing a useful sounding board for these ideas. Willie-Sue Haugeland co-implemented Walnut. Jim Donahue developed Hickory. Willie-Sue Haugeland, John Maxwell, and Jim Donahue have all helped with the Squirrel system. Mark Brown has maintained and elaborated the Cypress Storage level and was simultaneously the central designer and implementor of the Alpine file system which Cypress depends on for remote data storage.

Samuel Feldman, Dennis McLeod, Jim Donahue, Mark Brown, John Maxwell, Butler Lampson, William Kent, Ken Keller, Dushan Badal, and Peter Deutsch provided useful feedback on drafts of all or part of this document as it evolved over last year. Kathi Anderson and Subhana Menis helped prepare this report for publication.