Thoughts on Database Research
A User Perspective

Kurt Ingenthron
Computer Aided Design Consultant
143 Stoneway Trail
Madison, Alabama 35758

The future of computer aided design is in object oriented programming. If the database community hopes to participate in this future, it must reexamine some basic assumptions about the architecture of database systems Database system functionality can be added to object systems but if the performance cost is too high, it will never survive. Below are some suggestions for what can be done at a reasonable performance cost.

The object oriented paradigm provides a more practical approach to the partitioning of the global database than horizonal and vertical partitioning of relational tables. Each partition should itself be an independent database containing related data such as the geometry of a part or the spacial relationship of parts in an assembly. A meta-database would be used to control access to collections of these partitions. A collection of partitions comprise the database for a user's design session.

The overhead of traditional database transaction management is not acceptable for high performance CAD systems. With the partitioning scheme described above, transaction management can be performed at a partition/session granularity. Once the user has composed the collection of partitions, he has a single user database. There is no need for concurrency control or transaction logging except at the meta-database level. This type of transaction management can in fact be more functional than traditional transaction management, allowing for versioning, long transactions, integrity checking and archival.

Object oriented databases need a message model, not a data model. Any object which responds to the same messages as an object of "Duck" class (walk and quack) is, for all intents and purposes, a duck. An attempt to design a data model based on instance variables of an object or based on collections of objects of like class violates the data abstraction facilities of object oriented languages and diminishes their power. An attempt to implement a relational database system with an object oriented language yields a relational database system where you get abstract data types for free. It does not yield an object oriented database system.

For object oriented queries, the message is the media. A query can be transformed into an execution plan consisting of messages sent to database objects. Optimization decisions can be made by sending messages to referenced objects. Collection classes can be implemented for new access methods with cost and selectivity methods to provide optimization information In this way, the query language can grow with the application

Data representation is an important aspect of object oriented systems Most object systems are typeless in that all instance variables of an object are object references. For performance sake, object systems should provide enough of a type mechanism to allow simple data items (integers, floats, characters ...) to be represented in the form intrinsic to the machine. Methods can then be compiled for access to typed data.

In conclusion, object systems provide enormous potential for the development of CAD systems Performance influences the approach taken to an application. WYSIWYG publishing applications were not attempted until performance was adequate. Functionality is what sells CAD systems. Database system functionality can be added to object systems at a reasonable cost