Of Objects and Databases: A Decade of Turmoil

Michael J. Carey
David J. DeWitt

Presentation: Kyle Zeeuwen
Discussion: Anoop Karollil

Outline
- Introduction
- Note on Impedance
- Database System Toolkits/Components
- Persistent Programming Languages
- Extended Relational Database Systems
- OODB vs. ORDBMS: what’s difference?
- Predicting the Present

A note on impedances
- Object-Relational Impedance Mismatch:
  - Not just a problem getting an object into a DB representation
  - Access & Object Integrity also concerns
    - In OO we encapsulate info and only allow access via interfaces which are guaranteed to maintain representation invariants
    - RDBMSs have open query and update interface, hard to represent the invariants in RDBMS

Database Toolkits
- Like the µ-kernel of DBMS
- Provide ‘kernel’ of DBMS and toolkits to aid development and customization of most aspects of DB:
  - Query language
  - Optimizer
  - Access methods
  - Physical storage
- Motivation:
  - No single DBMS will satisfy requirements of all applications

Database Toolkits
- How does this address the impedance mismatch?
  - It doesn’t … it passes problem to the DBMS implementer to resolve
  - The impedance mismatch may or may not be an issue depending on the implementation of the DDL and DML

Persistent Programming Languages
- Extend the type system and programming model to add persistence to programs.
- An application developer specifies object persistence within code
- Applicable in domains where persistence is main concern as opposed to:
  - rich query support (optimization, expressiveness)
  - transaction management
Persistent Programming Languages

- How does this address the impedance mismatch?
  - By removing the DB there is no longer any mismatch. Problem solved...
  - But you lose many features of the DB
  - Whole host of new issues
  - Refer to previous discussion of Objectstore...

Casualties

- System Toolkits
  - Too much work to develop a DB from scratch, especially given reasonably good vendor solutions
  - No research or commercial products circa 1996

- Persistent Programming Languages
  - No commercial implementations, but their impact clear on OODB and object oriented client side wrappers
  - Still researched in 1996

Toolkits Example: EXODUS

- Project included storage manager, persistent language (based on C++), query optimizer generator
- Why did it fail?
  - Too much left to the implementer
  - The one thing people wanted to customize was already done (Client/Server Storage Manager) and 'got in the way'
  - Granularity of persistent storage language was not suitable

Object Oriented Database Systems

- Combine all RDBMS features with features of OO language to make new DBMS solution
- Similar to persistent programming
  - Difference lies in additional DB feature support (query language (i.e. OQL), indexing, transactions, etc.)

Object Oriented Database Systems

- How does this address the impedance mismatch?
  - Address the issue by providing tight integration between DB and programming language – no more mismatch
  - The OODB representation of an object identical or very similar to programming model

What features define an OODB?

*The Object-Oriented Database System Manifesto (1990):*

- Complex objects with unique identities
- Encapsulation
- Inheritance and Substitutability
- Late binding
- Extensible type system
- Persistence, concurrency, recovery
- Ad-hoc query support

And Optionally:

- Multiple inheritance
- Static vs. dynamic type checking
- Distribution, Long Transactions
- Version Management
Discussion

Was research into OODB driven solely by OO language needs, or can the OO paradigms of data abstraction and encapsulation enable a database system to store/manipulate data more efficiently as well?

Do you agree with the paper's characterization of ObjectStore as an OODB? Why or why not?

Extended Relational Database Systems

Provide an evolutionary path from current RDBMS

Extend RDBMS to allow definition of user defined types
  - Abstract Data Types (ADTs) – used as attributes
    - ADT specified in an external language
    - ADT methods can be used in queries
  - Row Types
    - add object-like properties to rows such as functions
    - Support Inheritance between row types
    - Multi-values attributes

What features define an ORDBMS?

Support for richer object structures and rules
Subsume RDBMS functionality
Open to subsystems (tools, middleware, etc.)

Extended Relational Database Systems

How does this address the impedance mismatch?
- Lessening the mismatch from the DB side: creating attribute types that more closely match application objects.
- Pushes some business logic to DB; the query can call functions on objects within the query predicates

What is the difference?

Key difference:
- "the top-most level of an object-relational database schema is still a collection of named relations"
- OODBMS has no relations

Evolution vs. Revolution
- ORDBMS build on RDBMS instead of scrapping relational model

OODBMS: Undecided Circa 1996

Huge amount of research papers, many research systems, many commercial products
What was holding it back?
- No consensus on feature set
- Not as mature as RDBMS systems
- Use of ODBC reduced impedance problem
- Vendors already began embracing RDBMS
**ORDBMS: Showing Promise**

- Several commercial offerings were available
- Adopting attractive OODB features
- Standardization work in SQL3
- Vendors offering ready-made ADT type packages
  - Authors underestimated this trend – this is how things are done today

**Visions of 2006**

- Commercial ORDBMS:
  - Full support for rich ADT’s (implemented in multiple languages)
  - Exports high level OO data model for use by middle-tier and client
- Commercial OODB:
  - Serving niche markets that demand high performance and seamlessness (NO mismatch)

**Discussion**

- Supposing that the OO programming paradigm was developed at the time of the development of the relational model, would we still have the relational model?
- Are you surprised by any of the authors predictions for 2006? Where did they go right, and where did they go wrong?
- Predict the future. What do you think OODB/ORDB/ RDB will be in 2016?

**Not Covered (If We Have Time)**

- OO Client Wrappers?
- CORBA, OLE, Java
- Middleware
- Research Challenges
  - Performance with objects (indexing, joining, selectivity predictions …)
  - Client Integration
  - Parallelizing ORDBMS
  - Legacy DB support