Object/Relational Access Layers
A Roadmap, Missing Links and More Patterns

Author: Wolfgang Keller

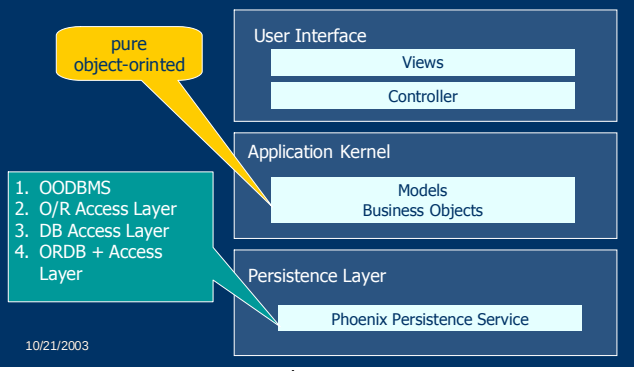
Contribution

- Provides a systematic roadmap of existing O/R mapping patterns.
- Provides some mapping patterns and links.

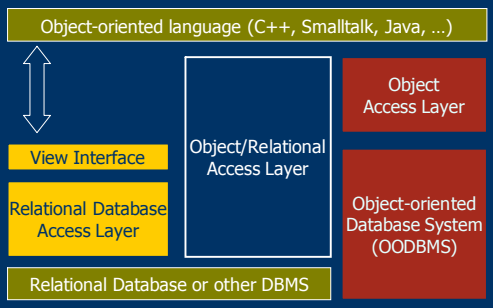
Outline

- Background
- roadmap.
- patterns

Architecture of Business System



Database Access Layer



Roadmap of the O/R Access Layer Patterns

A Architecting an O/R Access Layer		
B Designing the Object Layer	C Mapping Objects to Tables	
D Moving Attributes to and from the Tuple Layer	E Accessing Relational Databases	F Optimizing Performance
G Building the Access Layer		

pattern & pattern language

- A pattern is a recurring solution to a standard problem.
- Related patterns are woven together to form a informal pattern language.

Problem Forces Solution
Structure Example Consequences
Implementation Variants

Forces Driving the Language

- Functionality versus cost
- Separation of concerns versus cost
- Performance
- Flexibility versus complexity
- Legacy systems
- Application style (CAD, CASE, others)

Typical Functionalities for OODB

(1) Complex Objects	(9) Persistence
(2) Object Identity	(10) Secondary storage management
(3) Encapsulation	(11) Concurrency
(4) Types and Classes	(12) Recovery
(5) Class or Type Hierarchies	(13) Ad Hoc Query Facility
(6) Overriding, overloading and late binding	
(7) Computational Completeness	
(8) Extensibility	

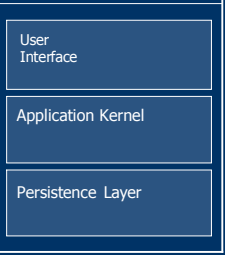
Architecting an O/R Access Layer

--Pattern List A

1. Layered Architecture for Business System
2. Two Layer Persistence Subsystem
3. Physical Views
4. Host Access

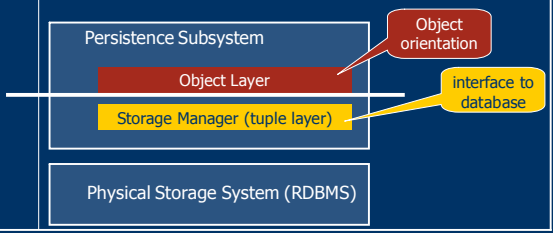
Pattern A1: Layered Arch. for Business Sys.

Problem	Which architecture is good for business system?
Solution	Three-layered architecture n User interface n Domain object layer n Persistence layer



Pattern A2: Two Layer Persistency Subsystem

Problem	Which structure is good for persistence subsystem?
Solution	Separation of concerns.

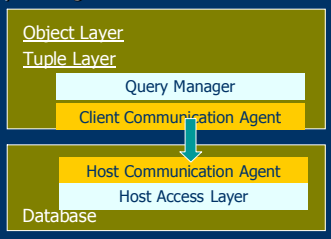


Pattern A3: Physical Views.

Problem	How to provide an easy to use interface to physical database tables?
Solution	n Encapsulate every table/view with a wrapper class. n Use wrapper classes to encapsulate overflow tables and other database optimization techniques. n Derive the wrapper classes from the protocol class to provide uniform interface.

Pattern A4: Host Access

Problem	How to link O/R access layer to host DB server?
Solution	Using a proxy-like agent.



Designing the Object Layer

Pattern List B

1. Object Identifier
2. Proxy
3. Object Manager
4. Transaction Object

Pattern B1: Object Identifier

Problem	How to represent an object's individuality in a relational database?
Solution	Assign each persistent object a key that will accompany the object from birth to death.

Note: Object ID is independent of its value.

Pattern B2: Proxy

Problem	How to prevent all related objects being loaded whenever you touch one object that has relations to many objects?
Solution	n Using a Smart Pointer (or Proxy) containing that object's identifier and a memory pointer. n In the initialization of a Proxy, the memory pointer is NULL.

Pattern B3: Object Manager

Problem	How to preserve object identity?
Solution	n Create a cache of objects per database client process. n Base the cache on a container that maps Object Identity to Proxies.

Pattern B4: Transaction Object

Problem	How to handle transactions at a user code level?
Solution	n Let each transaction be an object. n The transaction object has methods such as begin(), commit() and rollback().

Note: exception handling is a must.

Mapping Objects to Tables

Pattern List C

1. Single Table Aggregation and Foreign Key Aggregation
2. Foreign Key Association, Association Table
3. One Inheritance Tree One Table, One Class One Table, One Inheritance Path One Table, Objects in BLOB

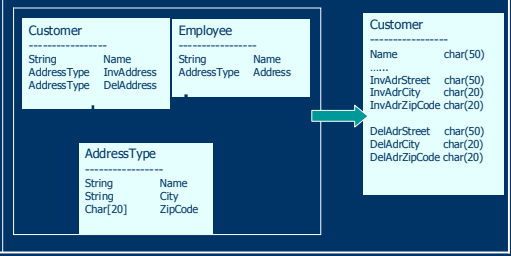
Mapping Objects to Tables

A Pattern Language

Author: Wolfgang Keller

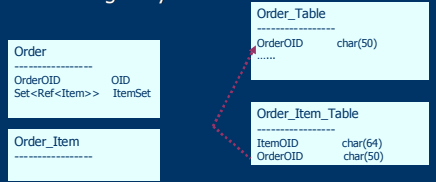
Pattern C1: Single Table Aggregation

Problem	How to map aggregation to tables?
Solution	Put to the same table.



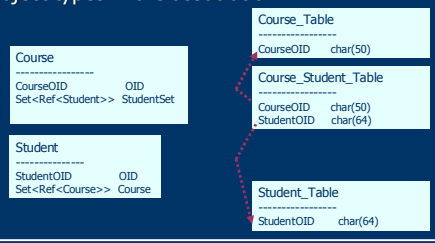
Pattern C2: Foreign Key Association

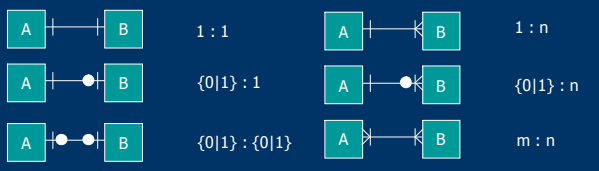
Problem	How to map 1:n association to tables?
Solution	Insert owner's OID to dependent table Make a foreign key



Pattern C3: Association Table

Problem	How to map n:m association to tables?
Solution	Use a separate table to contain the OID of the two object types in the association.

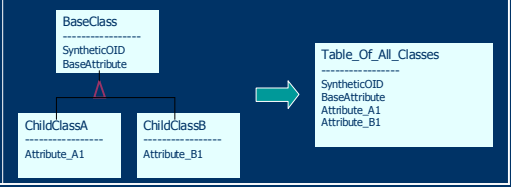




OID of A is required/optional in table B, duplication is allowed/not allowed

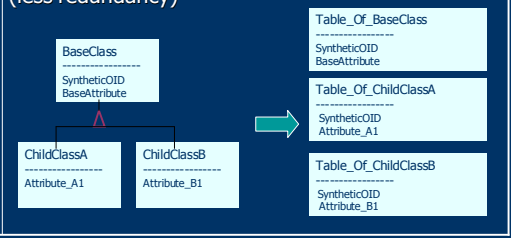
Pattern C4: One Inheritance Tree One Table

Problem	How to map inheritance classes to tables?
Solution	All in one table. (fast query with cost of overloading) (ambiguity: if attributes can be null)



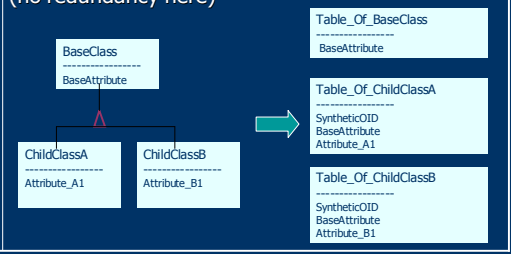
Pattern C5: One Class One Table

Solution	Map each class to a separate table. (fast query for basetype, slower for children) (less redundancy)



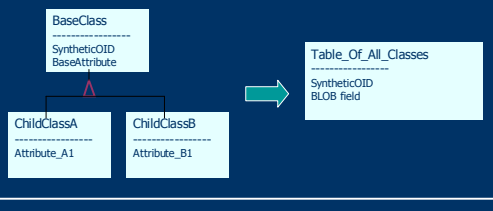
Pattern C6: One Inheritance Path One Table

Solution	Map each class to a separate table, add attributes. (fast query for children, slower for basetype) (no redundancy here)



Pattern C7: Objects in BLOBs

Solution	Use a table containing its syntheticOID and a variable length BLOB field.

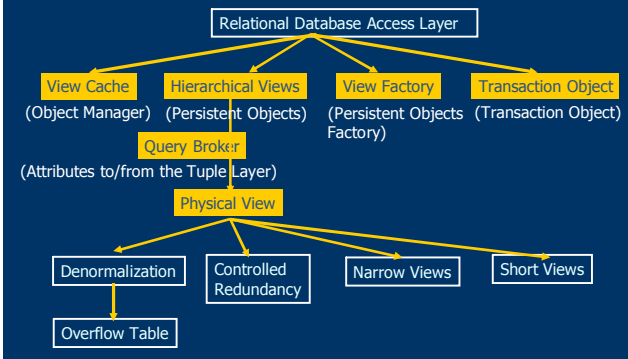


Moving Attributes to/from the Tuple Layer

Pattern List D

1. Multilayer Class
   upper layer and lower layer objects coupled
2. Class Broker
   one separate object for one upper layer objects.
3. Central Broker
   an object for all the upper layer objects.

Accessing Relational Database

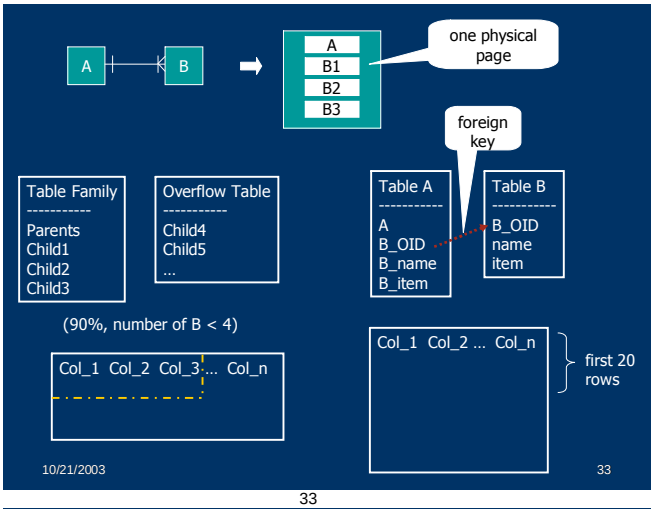


Optimizing Performance

Pattern List F

- Optimizing Table Structure and Queries
  - Denormalization
  - Overflow Table
  - Controlled Redundancy
  - Narrow Views
  - Short Views
- Optimizing General Design
  - Cluster Read
  - Bundled Write
  - Store for Forward
  - Flat File Write

(* reduce database traffic and disk I/O while keeping maintainability & reasonable cost)



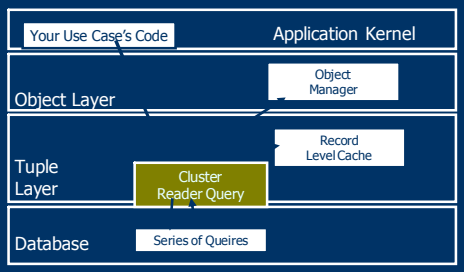
33

Pattern F1: Cluster Read

Problem	How do you provide high performance access to large chunks of data via an O/R access layer?

orthogonality?

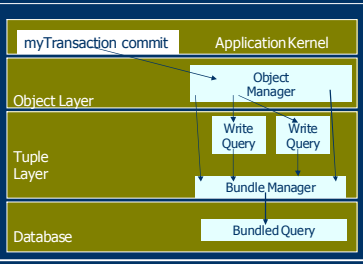
more clusters harder to maintain



34

Pattern F2: Bundled Write

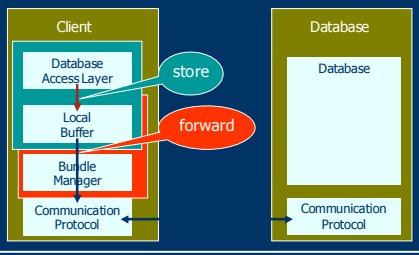
Problem	How do you speed up the process of writing dirty objects to the database?
Solution	



35

Pattern F3: Store for Forward

Problem	How do you shorten waiting time when there is too much data for transfer.
Solution	



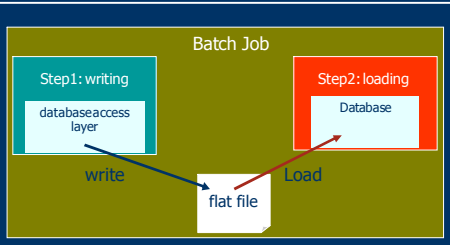
36

Pattern F4: Flat File Write

Problem	What if you can't wait for the database inserts when having too much data to write.

encapsulation?

only good for inserts, how about updates?



37

Building the Access Layer

Pattern List G

- Table Design Time
- Representing Objects as Tables

38

Questions & comments

?

39