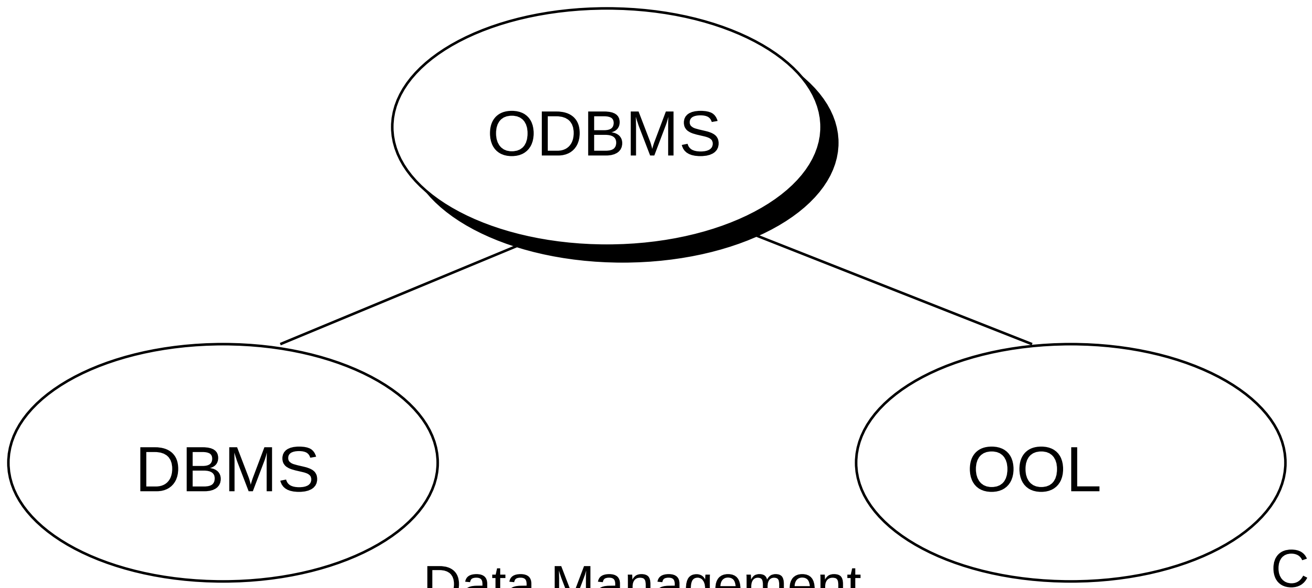
ODMG

C. Delobel

ODBMS History

- 1984 Maier and Copeland : birth of ODMBS concepts
- 1989 Manifesto : Atkinson et al
- 1991 First ODMG proposal
- 1993 ODMG standard

The Manifesto



Data Management
Disk
Indexing
Transaction
Security
Query

Class
Inheritance
Encapsulation
Object Idendity
Encapsulation
Extensibility

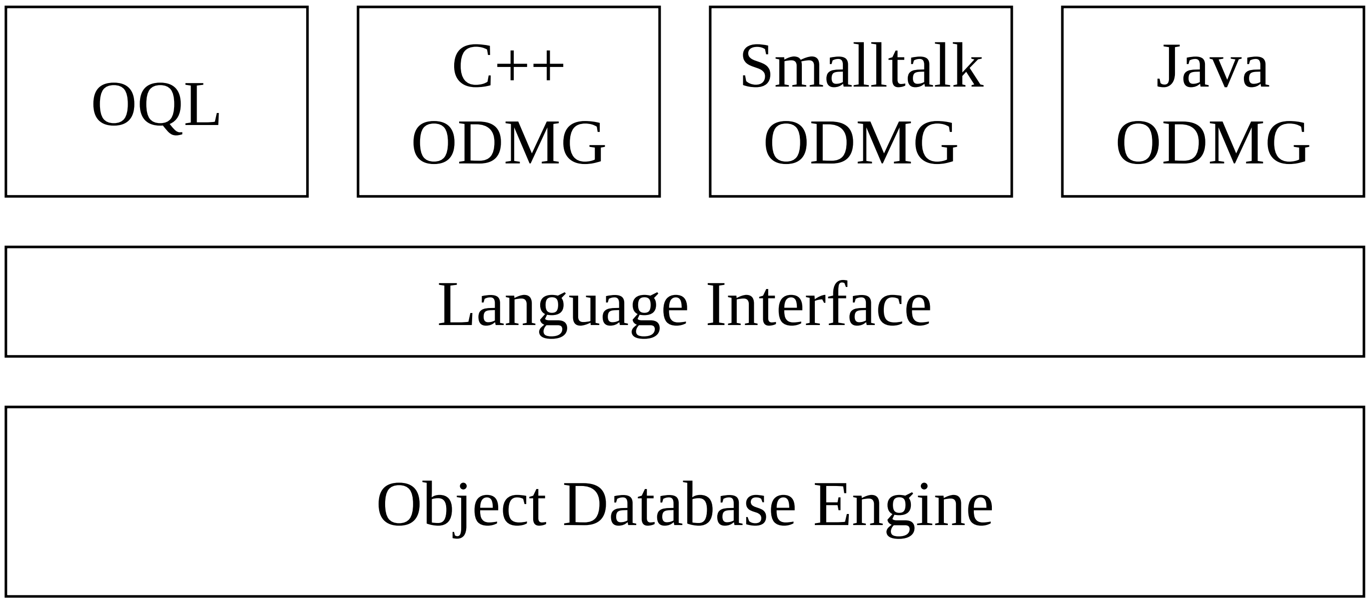
Object Data Base Management Group (ODMG)

- Founded in 1991.
- Members: Objectivty, Object Design, Ontos, O2 Technology, Versant.
- Reviewer members: Dec, HP, Intelletic, Itasaca, Poet, Servio, Texas.
- ODMG is a corporate association affiliated to OMG.
- 1993, The Object Database Standard, Morgan Kaufmann.

ODMG-93 Standard: Main Features

- Object data model
- Object Definition Language (ODL)
- Object Query Language (OQL)
- C++ binding
- Smalltalk binding
- Java binding

ODMG 93 architecture



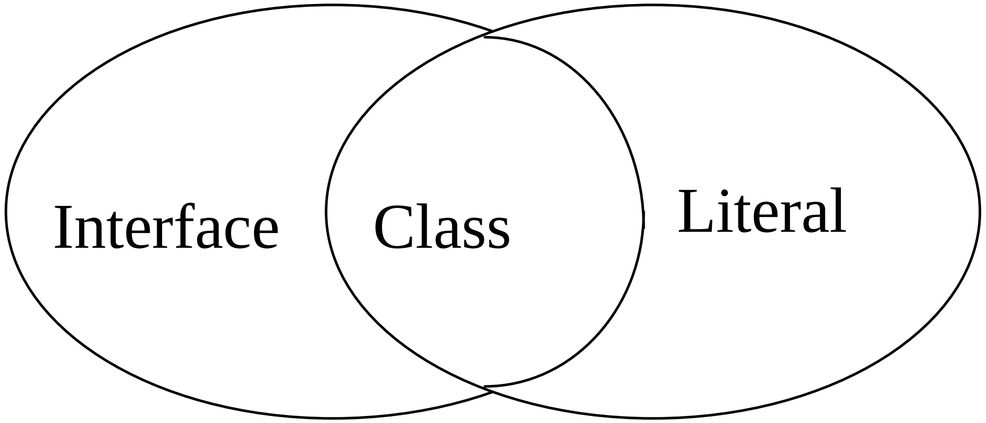
Object Data Model Principles

- Structural approach
  - ➤ building complex types (NF2)
  - ➤ distinction between object and value
  - ➤ class and/or type
- Behavioural approach
  - ➤ encapsulation principles (Abstract Data Types)
  - ➤ operations (constructor, destructor, read and write)
  - ➤ inheritance hierarchy
  - ➤ overloading and late binding

Basic properties

- Class or interface, method (operation), inheritance
- A set of atomic types
- Relationships
  - 1-1 relationships
  - 1-n, m-n relationships
- Collections (set, bag, array, list)
- Names for the entry points in the database

Class or interface



Abstract behavior
(operations)

Abstract state
(properties)

ODMG data type

- Object type
  - atomic objects are user defined
  - structured object (Date, Time, Interval...)
  - collection objetcs of the same type
- Litteral type (value)
  - atomic litteral (long, short, float, double, char, string, boolean, ...)
  - structutral litteral (Date, Time, ...) or user defined by a struct constructor
  - collection literals

Collections

- Generic collections parametrized by type: Collection<T>, where T can be an atomic type, a literal type or a class name.
- Collections are Set<T>, Bag<T>, List<T> and Array<T>

Metadata

- Metadata is descriptive information about the database schema
- Classes
- Property
  - attribute
  - relationship
- Operation (method)
- Exceptions

C. Delobel

Relationships

- Besides attributes of a class, we can name a relationships between clases.
- Relationships can be 1-1 or m-n.

ODL

- ODL is the Object Definition Language for the model
- ODL is not tied to a specific object language, its objective is to improve the portability across different language implementations
- An ODL type is specified by an interface or a class

ODL by example

```
class Person
        (extent people key name)
        {
        attribute String name;
        attribute Date dateofbirth
        attribute struct Adress { unsigned short number,
        string street, string city};
        relationship Person spouse inverse Person::spouse;
        relationship list<Person> children inverse Person::parent;
        relationship set<Person> parent inverse Person::children;
        operations ..........
} ;
```

Extent

- An extent is an optional property for a class. The extent contains all the instances of the class and its subclasses. The extent is maintained by the system.
- A key can be associated with an extent; it is a complementary way to identify an object or to retrieve an object. Object are identified by their object identity. A key is an integrity constraint to be checked by the system.

Object interface

```
interface Object {
        boolean same_as();
        Object copy();
        void delete();
};
```

Interface Collection

```
interface Collection : Object {
        exception ElementNotFound();
        boolean is_empty ();
        boolean contains (in any element);
        void insert_element ( in any element);
        void remove_element (in any element)
                raises (ElementNotFound);
        Iterator create_iterator();
        .........
}
```


C. Delobel

Interface Iterator

```
interface Iterator {
        exception NoMoreElement ();
        boolean at_end();
        any get_element () raises (NoMoreElement);
        void next_position raises (NoMoreElement);
}
```

Interface Set

```
interface Set : Collection {
        Set create_union (in Set aset);
        Set create_intersection (in Set aset);
        Set create_difference (in Set aset);
        boolean is _subset (in Set aset);
        ............
}
```

Interface List

```
interface List : Collection {
        exception InvalidIndex();
        any retrieve_element_at ( in unsigned long index)
                raises  (InvalidIndex);
        void remove_element_at (in insigned long index)
                raises (InvalidIndex);
        ...............
        List concat (in List alist);
        void append (in List alist);
}
```


C. Delobel

Persistancy

- Persistancy is orthogonal to types.
- Object can be transient or persistent.
- How an object becomes persistent depends on the programming language.
  - in C++, at creation time p= new(database) Person
  - in Smalltalk, by attachment to a persistent root.

Transaction

- Transaction obeys the classical properties: ACID

```
interface Transaction {
        void begin();
        void abort();
        void commit();
}
```

Interface Database

```
interface Database {
        void open(in string database_name);
        void close();
        void bind (in Object anobject, in string name);
        Object lookup (in string object_name);
}
```

Bibliography

- R. Cattell and al, Object Database Standard ODMG 2.0, Morgan Kaufmann, 1997.
- M. Atkinson and al, The Manifesto, Proceeding of the First DOOD Conference, 1990.
- F. Bancilhon, S. Cluet, C. Delobel, A Query Language for O2, in Building an Object-Oriented Database System Edited by F. Bancilhon, C. Delobel and P. Kanellakis, Morgan Kaufmann, 1992.