Connecting an object database system to the outside world

Francois Bancilhon

O_2 Technology

Contents

- global database picture
- object database systems
- connection to the rest of the world
  - object languages
  - relational databases
  - www

Global picture: four types of products four markets

- relational database systems
- object database systems
- relational-object database systems
- object-relational middleware

Relational database systems

- database systems: persistence, disk management, sharing, reliability, security, ad hoc querying
- supports the relational model
- supports SQL89, SQL92, SQL3
- Oracle, Sybase, Informix, CA, IBM & Microsoft

Object database systems

- database systems (all the standard features)
- support the ODMG data model, i.e., C++, Smalltalk and OMG
- offer persistent programming features: tight integration of the database and the programming language
- O_2, ODI, Versant, Objectivity, Gemstone and Poet

Object databases: target users

- provide database technology ot object users
- address object users who need database features
- object users:
  - C++, Smalltalk and Java programmers who need to interact with a database
  - object upper or lower case tool use who need a database target environment.
- database features:
  - persistence, large number of objects, shared object, safe & reliable objects
- object centric

Object-relational databases

- database systems (standard features)
- support the relational model
- extend it by some object features
- support SQL89
- extend it by some object features
- Illustra (Informix), Omniscience (Oracle?), UniSQL

Object-relational databases: target users

- relational users
- bring them the benefits of object technology
  - complex objects, ADTs, inheritance, etc.
- respect their legacy applications, culture, expertise
- relation centric

Relational-object middleware

- bridge between relational system and object environments (languages or lower case tools)
- relation or object centric
  - relation centric: start from the relational schema, derive an object view
  - object centric: start from an object view, run it on top of a relational engine
- Persistence, Subtleware, Rogue Wave

Relational-object middleware: target users

- users with existing relational database system who want to use object tools.
- object development with legacy systems.

ODMG

- define a standard for object database application portability
- consortium of Object database vendors (Gemstone, O_2, Objectivity, ODI, Poet, UniSQL, Versant)
- reviewing members (TI, HP, Sybase, EDS, Anderson Consulting, Fujitsu, Microsoft, etc.)
- academic members (DeWitt, Maier, Moss, etc.)

ODMG: History

founded: September 1991

first version of the standard: October 1993

ODMG incorporated: January 1994

second version of the standard 1.1: April 1994

first compliant products: October 1995

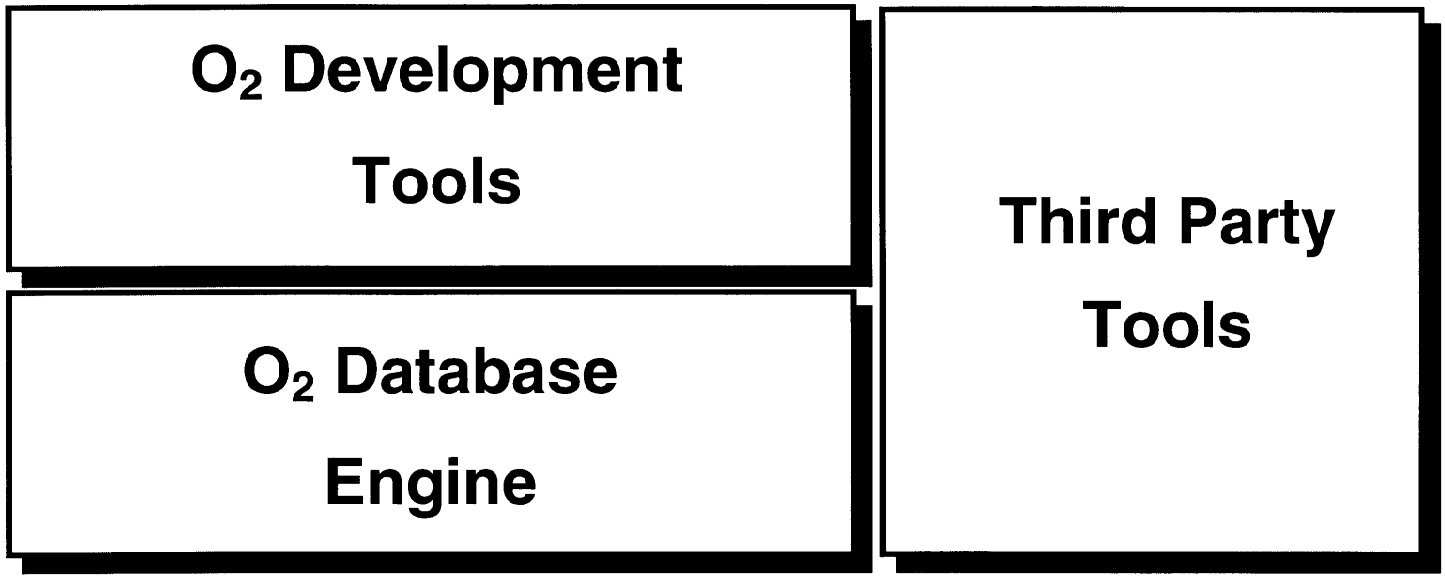
third version of the standard 1.2: December 1995

ODMG Standard

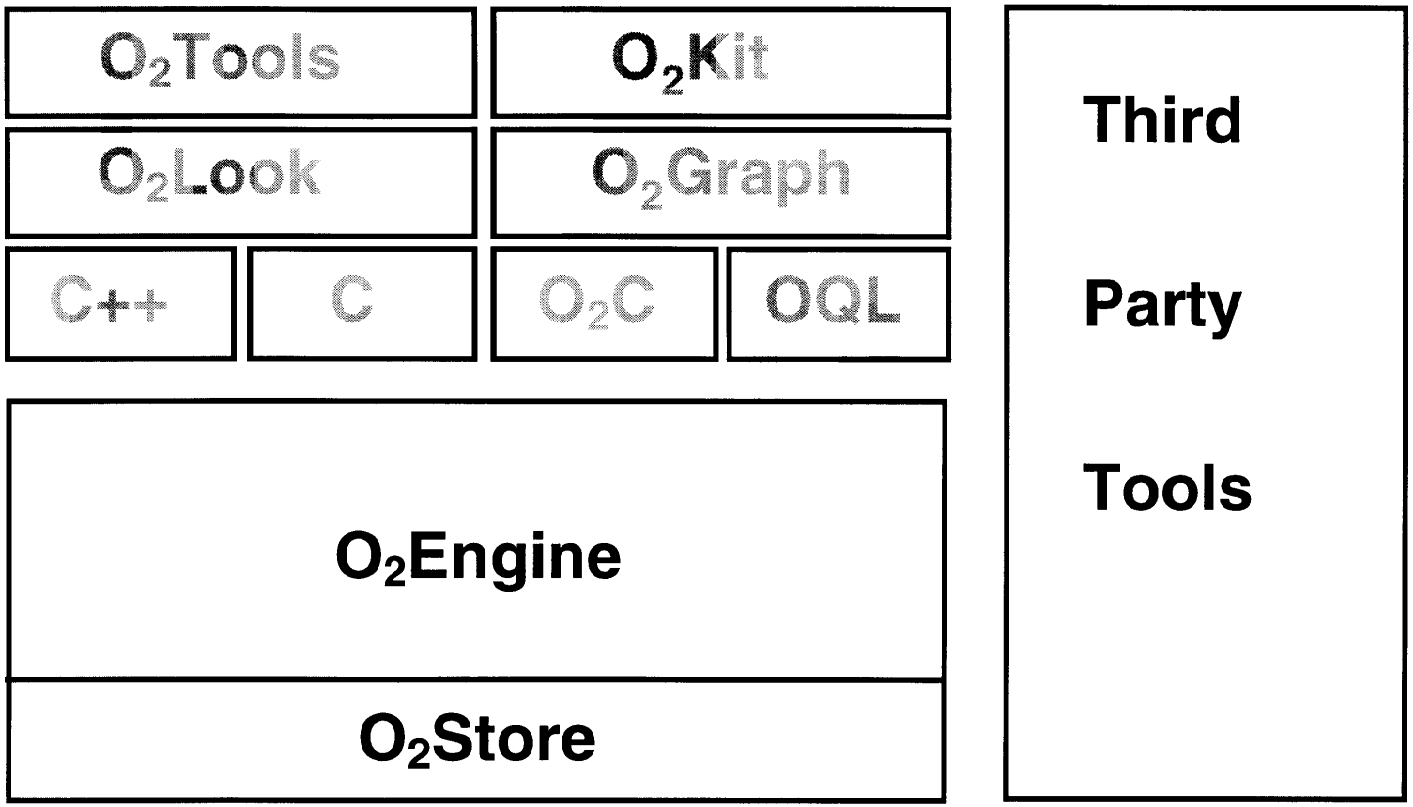
- Object model (extension of OMG)
- ODL (extension of IDL)
- OQL (from O_2SQL)
- C++ bindings
- SmallTalk bindings

- + more stuff: meta schema, Java, etc.)

O_2 product line



O_2 product line



Connections to the rest of the world

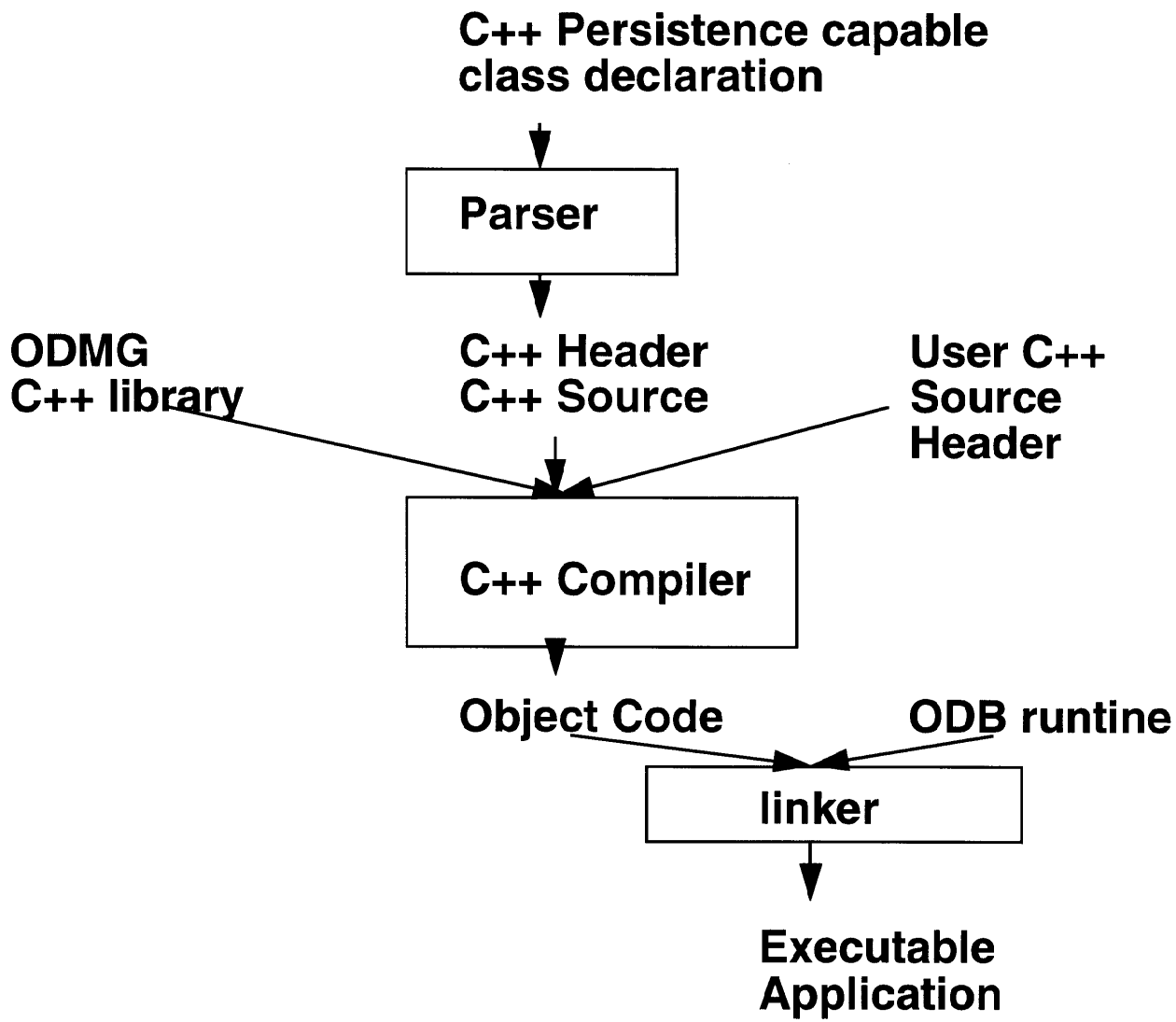
- languages
  - C++
  - Smalltalk
  - Java
- WWW
- relational databases

ODMG C++ interface

- persistence model: persistent capable classes, static creation model, persistent pointers
- data model (C++, 2-way relationships, collections, base classes)
- integration with OQL

ODMG C++ implementation

- 2 different representations of the objects: database and main memory
- maintain 2 copies of the object (2 caches)
  - the database copy
  - the C++ copy
- most robust, safer, good performance


C++ Persistence capable
class declaration
Parser
ODMG
C++ library
C++ Header
C++ Source
User C++
Source
Header
C++ Compiler
Object Code
ODB runtine
linker
Executable
Application

O_2Web: a WWW Connection Tool

- connects an O_2 database system to the WWW
- O_2 + O_2Web is a tool to build and administer WWW servers
- a very good match between two technologies (WWW and object databases)

Why use a database to set up a WWW Server (database vs. files)

- large servers will use large numbers of files and large numbers of links => maintenance and evolution is a problem
- performance
- administration
- multiple actors
- data modeling

Why use an object database to set up a WWW server (object vs. relational)

- multimedia data
- data and behavior
- network data model
- performance

O_2Web: mapping an object database on the WWW

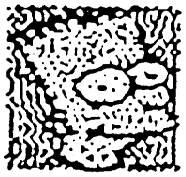
- presenting objects on the net
- addressing objects on the net

O_2Web: standard object presentation

- each O_2 object is translated into HTML
- each O_2 object has a standard presentation
- depends on the structure (atoms, collections, tuples, objects, date, text, bitmap, image) as specified in the database schema

```
class Entry public type
tuple(name : string,
      photo : Image,
      address : string,
      phone : string,
      e_mail : string,
      previous : Entry,
      next : Entry,
      up : Directory)
end;
method public title:string in class Entry;
```

O2Web

- **name** : Bart

- **photo** :
- **address** : Bart's address
- **phone** : 435444343
- **e_mail** : bart@site.domain
- **previous** :
- **next** : Grampa
- **up** : Personal PhoneBook

O_2Web: customized object presentation

- the on-screen presentation can be customized
- just redefine the translation method of the class
  - for the application
  - for the class
  - for the object

Netscape Grampa

File Edit View Go Bookmarks Options Directory Windows Help

Back Forward Home Reload Images Open Print Find Stop

Go To:

Personal PhoneBook Directory

[top] [up] [previous] [next]

Granpa

Address : Granpas's address

Phone : 435444343

Email : granpa@site.domain

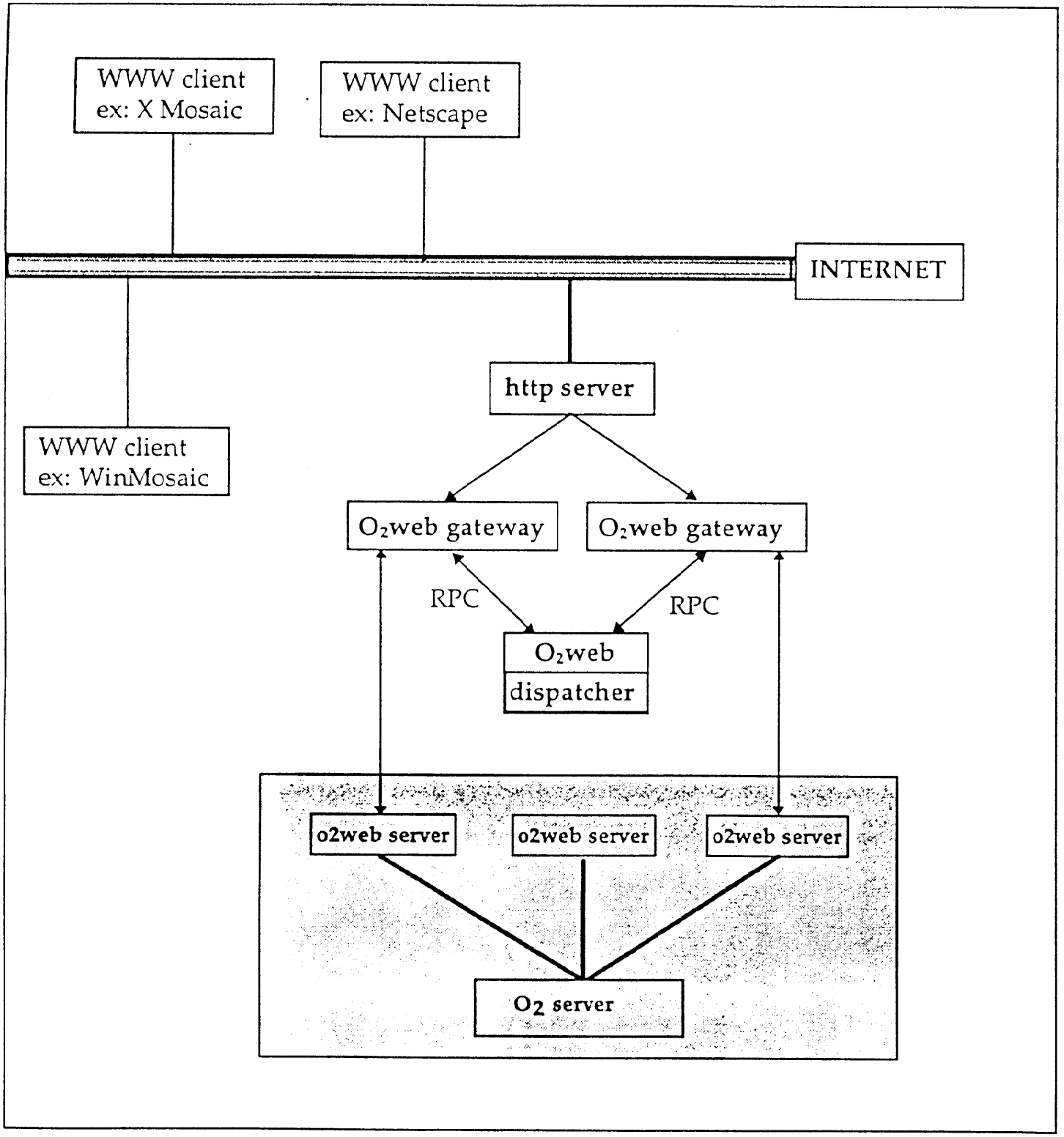
This demo is built on top of OZWeb - The O2 gateway of O2Technology

O_2Web: object addresses

- every object in the database can be accessed by a URL
- the URL contains
  - the O_2 System address
  - the OQL query accessing the object in the database
- WWW addressing requires both associative and navigational access. OQL is the only query language offering these features.

O_2Web: architecture

- includes
  - O_2Web Gateway
  - O_2Web Dispatcher
  - O_2Web Server
- requires
  - a WWW server (e.g. Nescape server)
  - a WWW client (e.g. Netscape browser)
  - an O_2 System (development or run time)


WWW client
ex: X Mosaic
WWW client
ex: Netscape
INTERNET
WWW client
ex: WinMosaic
http server
O2web gateway
O2web gateway
RPC
RPC
O2web
dispatcher
o2web server
o2web server
o2web server
O2 server

O_2Web: operation

- Web client sends a URL in HTTP format (the URL contains an OQL query)
- HTTP server passes the query to O_2Web Gateway
- O_2Web Gateway connects to O_2Web Server
- O_2Web Server runs the query, transforms the result of the query into HTML
- HTML data is sent to the Web client

Connection to relational databases

- O_2DBAccess (interoperability tool)
- migration tool
- SQL interface
- ODMG on relational

O_2DBAccess

- specify the remote site hardware
- specify the remote site software
- specify the network
- the tool establishes a link to the relational system

O_2DBAccess

- class library: connection, session, context
- establish a connection
- start a session (begin/commit/abort transaction)
- send an SQL query or statement
- map the result onto O_2 objects (of the relation type)

Connecting O_2

```
o2 Connection host = new Connection;
o2 Session session;
host->connect("where", "username", "pswd");
session = host->logon
     ("dbname", "dbusername/pswd").session;
....
host->logoff(session);
host->disconnect;
```

In O_2

```
class Employees
     list(tuple(name: string,
                dept: integer,
                salary: real)) end;
```

In the RDB

```
create table Emp(ename character(16)
                   edeptno integer
                   esalary real
                   ejob: character(32))
```

Connecting O_2

```
...
o2 Session session;
o2 integer status;
o2 string stmt =
     "select ename, esalary, edeptno
      from emp
      where deptno=15 and job='sales'"
o2 Employees res = new Employees;
status = session->sqlquery(res, stmt)
...
```

Object-relational migration tool (*)

- specify a relational schema
- specify an object schema
- specify the mapping between the 2 schemas (in OQL)
- type check the consistency
- generate a loader
- migrate the database

SQL interface to O_2 (*)

- benefits :
  - ODBC interface,
  - marketing,
  - access from relational environments
- start from an ODMG object schema
- generate a relational view on top of the object schema
- two strategies:
  - materialize the view and run SQL on it
  - translate SQL into OQL queries on the object database

SQL interface to O_2

- start from an object schema
- generate a default relational view
- customize the default view through a configuration file
- translate SQL queries on the view into OQL queries on the object database

Object to relational translation: restrictions on the object schema

- every class is of a tuple type
- every class has an extension (or a way to compute it)
- every class has an oid method

```
schema document;
class Article type tuple(
   title:string,
   authors:set(Author),
   sections:list(Section)
   date:Date)
end;

class Author type tuple(
   name:string,
   address:tuple(institute:string,country:string)
   email:string)
end;

class Section type tuple(
   title:string,
   contents:string)
end;

class Date type tuple(
   day:integer,
   month:integer,
   year:integer)
end;

name Articles:set(Article);
```

Object to relational translation

- map each object class to a table with the oid
- if flat tuple finished
- if embedded tuple, just flatten
- if relationship to another object, put a foreign key
- if set attribute, generate an intermediate table
- if list attribute, generate an intermediate table with index number
- map ODMG base types to SQL2 base types

```
CREATE SCHEMA document

CREATE TABLE Article( key CHARACTER(16) title CHARACTER(32)
                      date_key CHARACTER(16) PRIMARY KEY ( key )
                      FOREIGN KEY ( date_key ) REFERENCES Date )

CREATE TABLE Author( key CHARACTER(16) name CHARACTER(32)
                     institute CHARACTER(32) country CHARACTER(32)
                     email CHARACTER(32) PRIMARY KEY ( key ) )

CREATE TABLE Date( key CHARACTER(16) day INTEGER month INTEGER
                   year INTEGER
                   PRIMARY KEY ( key ) )

CREATE TABLE Section( key CHARACTER(16) title CHARACTER(32)
                      contents CHARACTER(32) PRIMARY KEY ( key ) )

CREATE TABLE Article_authors( Article_key CHARACTER(16)
                              authors_key CHARACTER(16)
                              FOREIGN KEY ( Article_key )
                              REFERENCES Article
                              FOREIGN KEY ( authors_key )
                              REFERENCES Author )

CREATE TABLE Article_sections( Article_key CHARACTER(16)
                               index INTEGER
                               sections_key CHARACTER(16)
                               FOREIGN KEY ( Article_key )
                               REFERENCES Article
                               FOREIGN KEY ( sections_key )
                               REFERENCES Section )
```

Object to relational mapping: customizing the relational schema

- change the names of tables
- change the names of columns
- hide attributes and relationships (projection)
- hide classes
- declare methods as attributes
- declare extension as OQL queries

Object to relational mapping: SQL to OQL translation

- from the schema mapping and the SQL query
- generate the OQL query
- optimize it

ODMG compliant C++ interface on Relational Systems (*)

- allows to store and retrieve C++ objects in a relational database
- ODMG standard (will run directly any ODMG compliant C++ program on top of the relational system)
- benefits: same application runs on O_2 and Oracle

Functional architecture

C++ Application

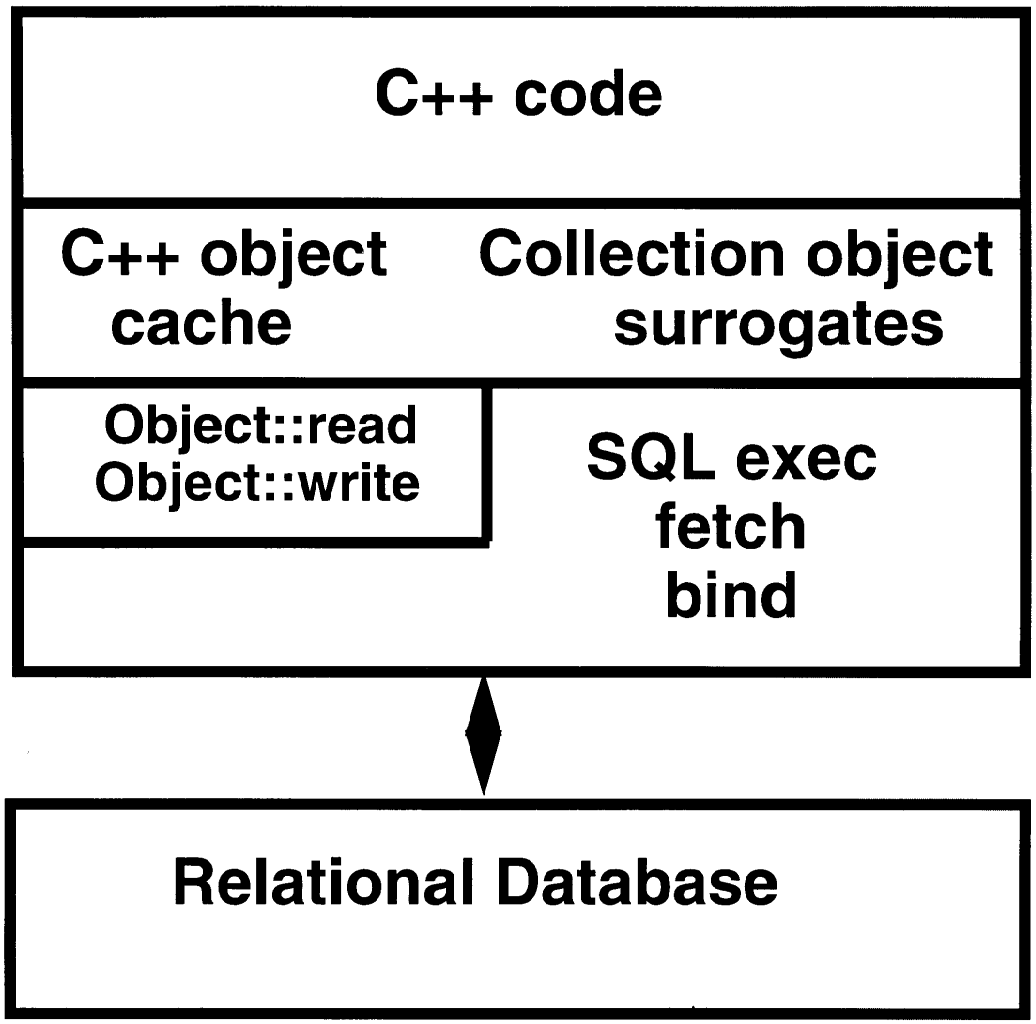
C++ ODMG Run time

C++/RDB Object Mappings

Relational DB client interface

RDB Client/Server Protocol

Relational DB Server



C++/Relational System run time: object manipulation

C++ object cache maintained transparently through the Ref technique (ODMG standard)

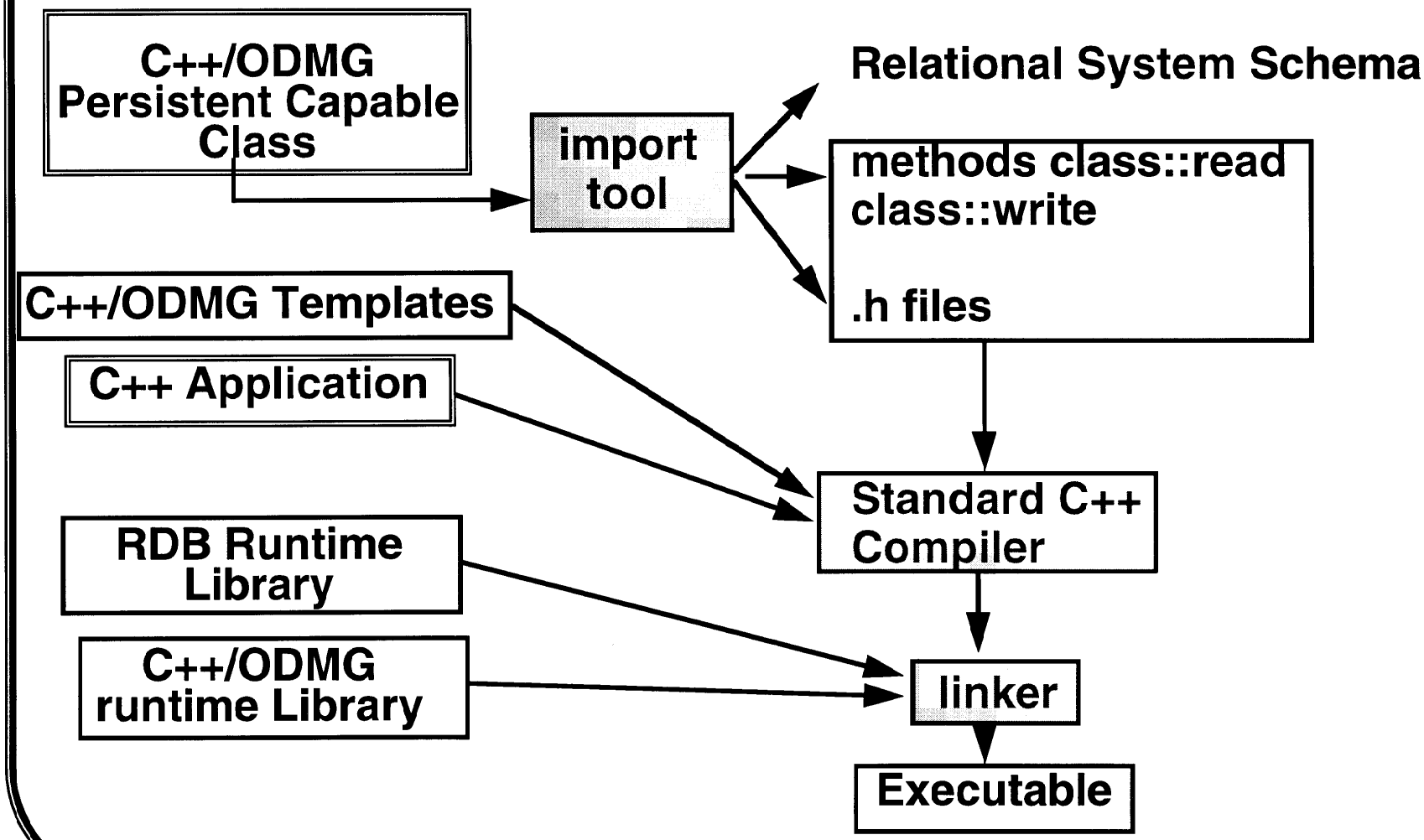
- Object Fault:

```
Ref<Person> p;
...
p->name;
...
```

overload the -> of p->name:
if p is not in cache then p->read

- Commit or Swap (cache is full): p->write

Development environment



More connections

- Corba integration (O_2 and Orbix)
- XA support

Conclusion

- performance (engine)
- good integration with programming languages (C++, Smalltalk, Java)
- good connections to the rest of the world