

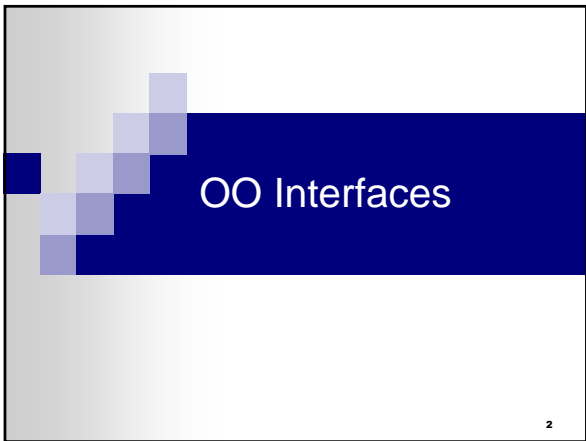
COMP67321

Introduction to Object-oriented Databases

Goran Nenadic

School of Computer Science
University of Manchester

1



OO Interfaces

2

ODMG interfaces

- An interface describes visible attributes, relationships and operations
 - it is a specification of the abstract **behaviour** (operation signatures) of an object type
- Used to specify abstract operations that can be inherited by classes or other interfaces
 - “**behaviour inheritance**”
- State properties of an interface (i.e. its attributes and relationships) cannot be inherited
 - objects cannot be *instantiated* from an interface

3

continued

ODMG interfaces

- All objects inherit the basic Object interface which provides **basic operations**
 - create a new copy
 - delete the object
 - compare (identities) of objects

4

Interface definition

- Interface describes visible attributes, relationships and operations
- Properties of an interface (i.e., its attributes and relationships) cannot be inherited
 - objects cannot be *instantiated* from an interface
- Behaviour inheritance denoted using ‘:’


```
interface Date:Object
```

5

continued

Interface definition

- All objects inherit the basic Object interface


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

basic operations:

 - create a new copy
 - delete the object
 - compare (identities)

6

Interface definition – example 1

```
interface Date:Object {
  enum weekday{sun,mon,tue,wed,thu,fri,sat};
  enum Month{jan,feb,mar,....,dec};
  unsigned short year();
  unsigned short month();
  unsigned short day();
  ...
  boolean is_equal(in Date other_date);
  boolean same_as(in Date other_date);
  ...
};
```

7

Interface definition – example 2

```
interface GeometryObject {
  attribute enum Shape{ Rectangle, Triangle, Circle} shape;
  attribute struct Point{ short x, short y} reference_point;
  float perimeter();
  float area();
  void rotate(in float angle_of_rotation);
};
class Circle : GeometryObject
{
  attribute struct Point{ short x, short y} reference_point;
  attribute short radius;
};
```

8

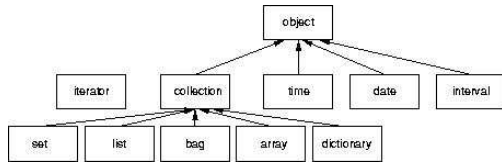
continued

Interface definition

- No objects can be created based on GeometryObject (as it is an interface)
- Several classes can inherit an interface
 - but, **only operations** are inherited, not properties (attributes, relationships)
- Multiple inheritance of interfaces allowed by a class or by an interface
 - class can inherit from at most one class (via extends) and from zero or more interfaces

9

ODMG interfaces (built-in)



inheritance hierarchy for built-in interfaces of the Object model

10

Built-in interfaces for collections

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

11

Built-in interfaces: iterators

- iterators are one of the basic interfaces in the object model
- iterates over a collection
- provides methods such as
 - at_end()
 - reset()
 - get_element()
 - next_position()

12

example

Built-in interfaces for collections

- each collection type (set, list, bag, array, dictionary) provides additional methods

```
interface Set: Collection {
    Set    create_union(in Set other_set);
    ...
    boolean is_superset_of(in Set other_set);
};
```

- Note: look at the methods for other collections

13

Database interface

```
interface Database{
    void    open(in string db_name);
    void    close();
    void    bind(in any some_obj, in string obj_name);
    Object  unbind(in string obj_name);
    Object  lookup(in string obj_name)
            raises(ElementNotFound);
    ...
};
```

```
interface DatabaseFactory{
    Database new();
};
```

14
