# Terminology

Noël PLOUZEAU
IRISA/ISTIC





# A glossary

- The Internet and the SE domain provides several meanings for a given word
- For clarity we need one word for one concept
- ACO uses mainly definitions from UML, including for coding





# Interface/implementation

- This separation is a critical key concept in software engineering
- Object-oriented software has a strong support for it

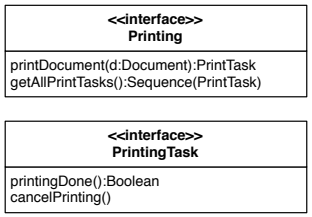


## Interface concept

- An interface
  - includes all what is needed to ask for a service
  - and nothing more












## Java interface

```
interface Printing {
  public PrintTask printDocument(Document d);
  public List<PrintTask> getAllPrintTasks();
}
```

## Operations

- An interface (UML, Java, etc) contains only operations
- To request a service from an object one needs nothing more: no notion of implementation



## Rule #52

- In design (UML models) and code
  - refer to objects through interfaces
- This applies to
  - parameters
  - attributes
  - return types



## Class

- In most cases it is an implementation of an interface
- When accessing an object you don’t need to know its class, only its interface
  - This is an important point

## Attributes

- Internal state of an object
- A good design must preserve encapsulation (see rule #14 later)



## Message

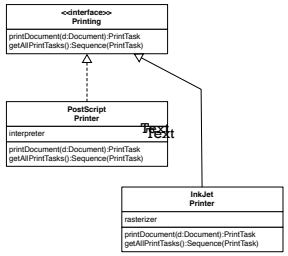
- An object o1 requests a service from another object o2
  - by sending a message
  - and waiting (or not) for a reply
- Messages are native in the UML, not in Java



## Method

- An implementation of an operation (usually code)
- Object-oriented languages allow for 0, 1 or more implementations for an object

# Operation != Method




2 operations, 4 methods





## When o2 receives a message

- It looks for what to do
  - usually this means calling an operation with the same name
  - in UML we can write automata





## When an operation is called

- The runtime looks for a method for this operation
  - if one is defined in o2's class it is run
  - otherwise the runtime looks in the ancestor class(es)

## Abstract/concrete

- A concrete class has at least one method for each operation
- An abstract class with at least one operation without a method



## In Java, C#, C++

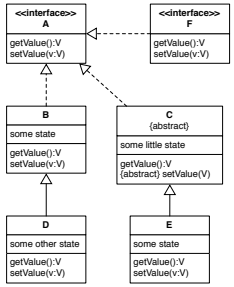
- A method search is always successful
  - because you cannot instantiate an object of an abstract class
  - this means that each operation has at least one implementation (*ie* method)



## Inheritance

- Two forms:
  - operation inheritance
  - method inheritance
- Really different
  - this is why it is wise to use two different terms (operation and method)

# Example of inheritance (UML)








# References

- Java spec
  - http://docs.oracle.com/javase/specs/
- UML spec
  - http://www.uml.org





# References (cont'd)

- Bloch
  - BLOCH, Joshua. Effective java. Addison-Wesley Professional, 2008.
- Design patterns
  - GAMMA, Erich, HELM, Richard, JOHNSON, Ralph, et al. Design patterns: Abstraction and reuse of object-oriented design. Springer Berlin Heidelberg, 1993.