Design Patterns

MSc in Communications Software

Produced by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

Prototype

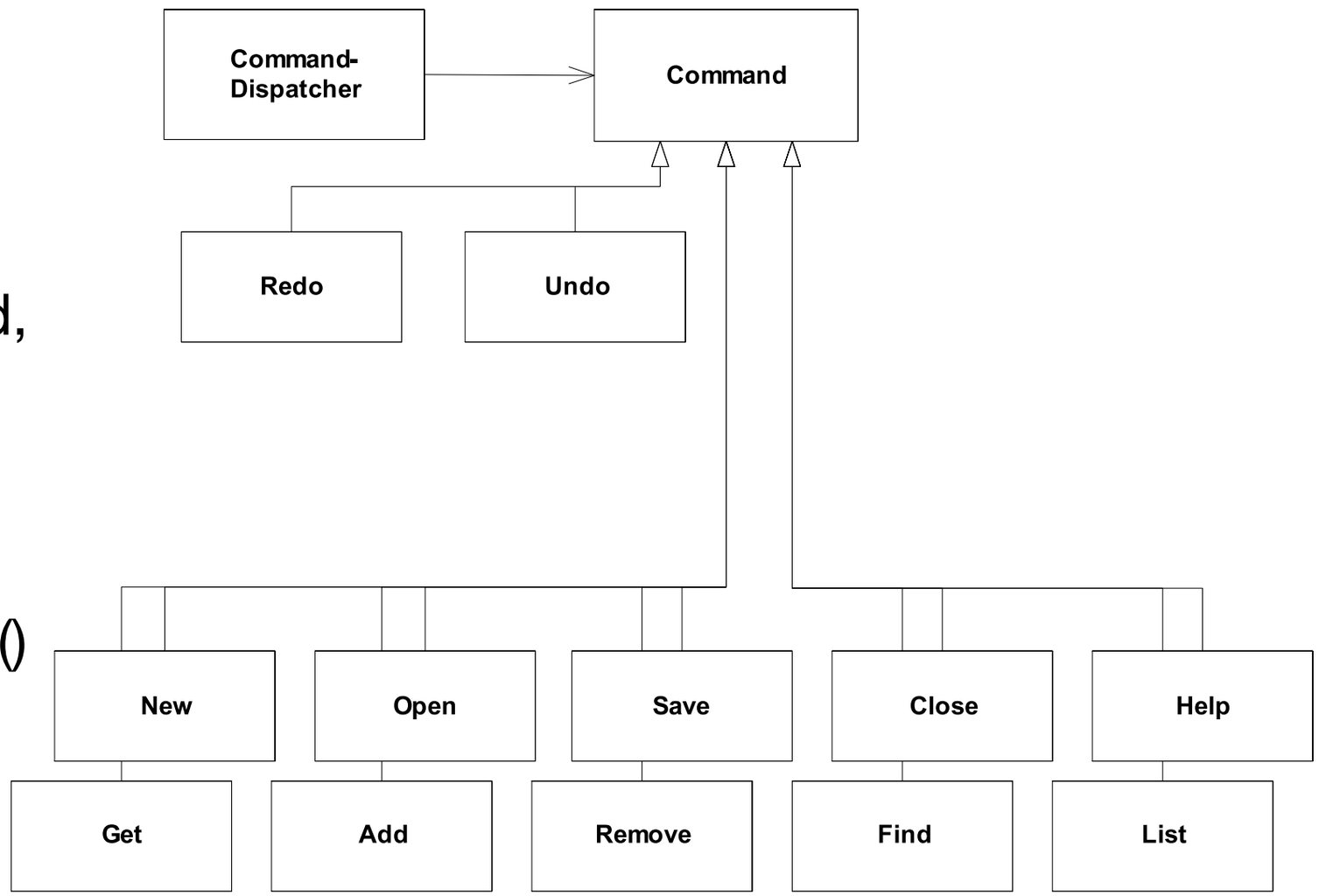
Design Pattern

Intent

- Specify the kinds of objects to create using a prototypical instance, and create new objects by copying this prototype

Motivation

- A CommandDispatcher is pre-loaded with a set of objects.
- These objects are all of type Command.
- When a Command is executed, a copy is to be made and this copy used for undo/redo mechanisms.
- The Command contain a copy() method for this purpose.
- Each Command implementation will provide its own implementation, copying itself on demand.



Applicability

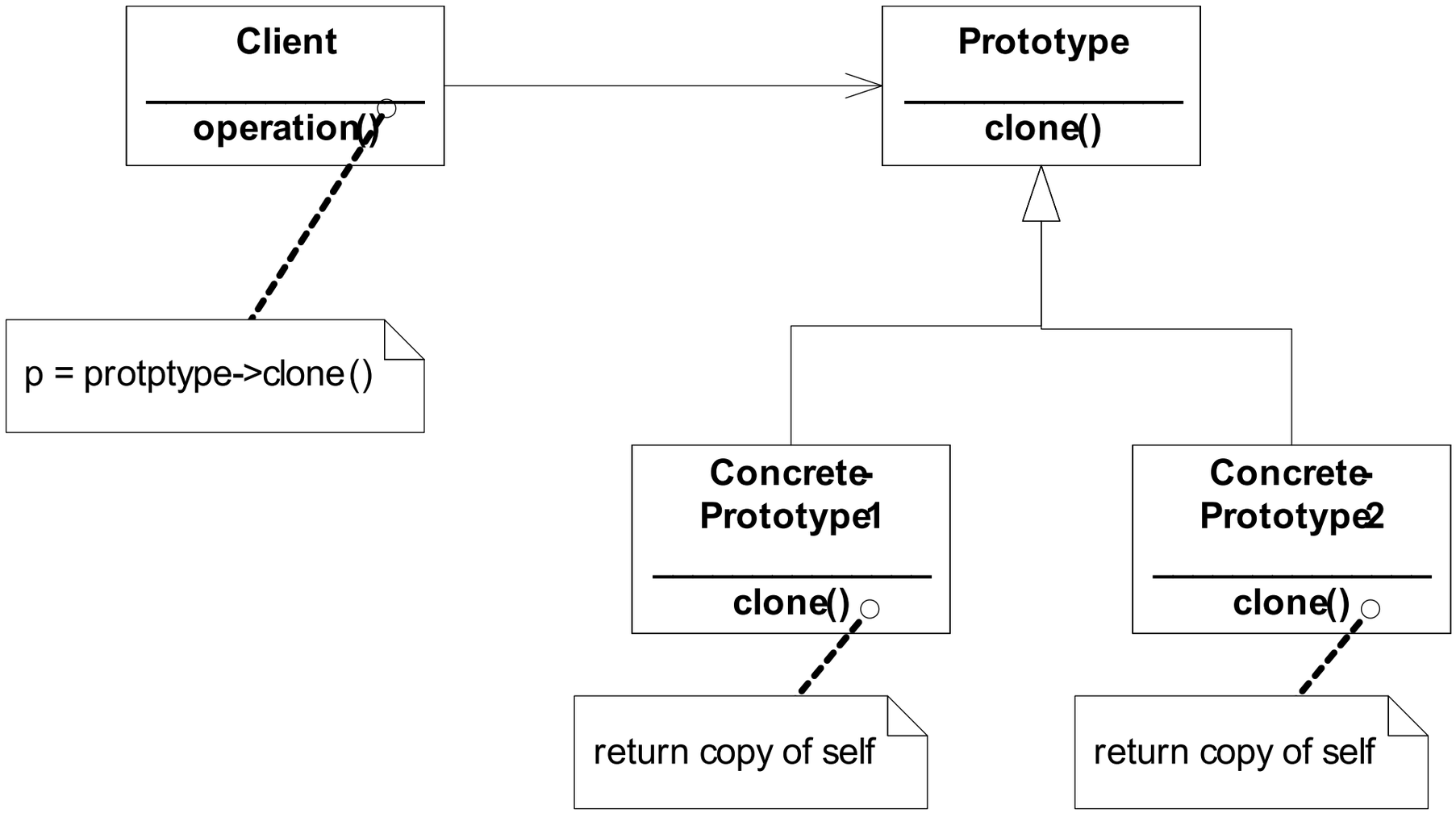
- Use the Prototype pattern when a system should be independent of how its products are created, composed, and represented; and

- when the classes to instantiate are specified at run-time, for example, by dynamic loading; or

- to avoid building a class hierarchy of factories that parallels the class hierarchy of products; or

- when instances of a class can have one of only a few different combinations of state. It may be more convenient to install a corresponding number of prototypes and clone them rather than instantiating the class manually, each time with the appropriate state.

Structure


Client
operation()
Prototype
clone()
p = protptype->clone()
Concrete-
Prototype1
clone()
Concrete-
Prototype2
clone()
return copy of self
return copy of self

Participants & Collaborations

- Participants
  - Prototype (Command)
    - declares an interface for cloning itself.
    - ConcretePrototype (Add, Remove etc…)
    - implements an operation for cloning itself.
- Client (CommandDispatcher)
  - creates a new object by asking a prototype to clone itself.
- Collaborations
  - A client asks a prototype to clone itself.

Consequences (1)

- Adding and removing products at run-time.
    - Prototypes let you incorporate a new concrete product class into a system simply by registering a prototypical instance with the client.
- Specifying new objects by varying structure.
    - Many applications build objects from parts and subparts.
    - Editors for circuit design, for example, build circuits out of subcircuits.
    - For convenience, such applications may enable creation of complex structures via cloning of existing set of pre-configured objects (e.g to use a specific subcircuit)

Consequences (2)

- Reduced subclassing
    - Factory Method often produces a hierarchy of Creator classes that parallels the product class hierarchy.
    - The Prototype pattern lets you clone a prototype instead of asking a factory method to make a new object.
    - Hence you don't need a Creator class hierarchy at all.

Implementation (1)

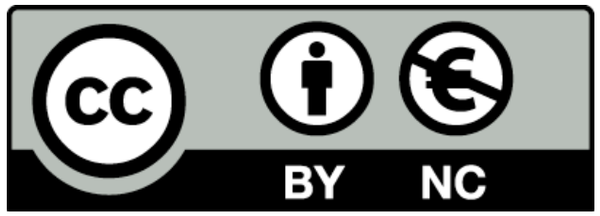
- Using a prototype manager.
  - When the number of prototypes in a system isn't fixed (that is, they can be created and destroyed dynamically), keep a registry of available prototypes.
  - A client will ask the registry for a prototype before cloning it. We call this registry a prototype manager.
  - A prototype manager is an associative store that returns the prototype matching a given key.
  - It has operations for registering a prototype under a key and for unregistering it.
  - Clients can change or even browse through the registry at run-time. This lets clients extend and take inventory on the system without writing code.

Implementation (2)

- Implementing the Clone operation.
  - The hardest part of the Prototype pattern is implementing the Clone operation correctly.
  - It's particularly tricky when object structures contain circular references.
  - Most languages provide some support for cloning objects
  - e.g. Java the Cloneable interface .
  - "shallow copy versus deep copy" problem - does cloning an object in turn clone its instance variables, or do the clone and original just share the variables?

Implementation (3)

- Initializing clones.
    - While some clients are perfectly happy with the clone as is, others will want to initialize some or all of its internal state to values of their choosing.
    - You generally can't pass these values in the Clone operation, because their number will vary between classes of prototypes.
    - Some prototypes might need multiple initialization parameters; others won't need any.
    - Passing parameters in the Clone operation precludes a uniform cloning interface.



Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit