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

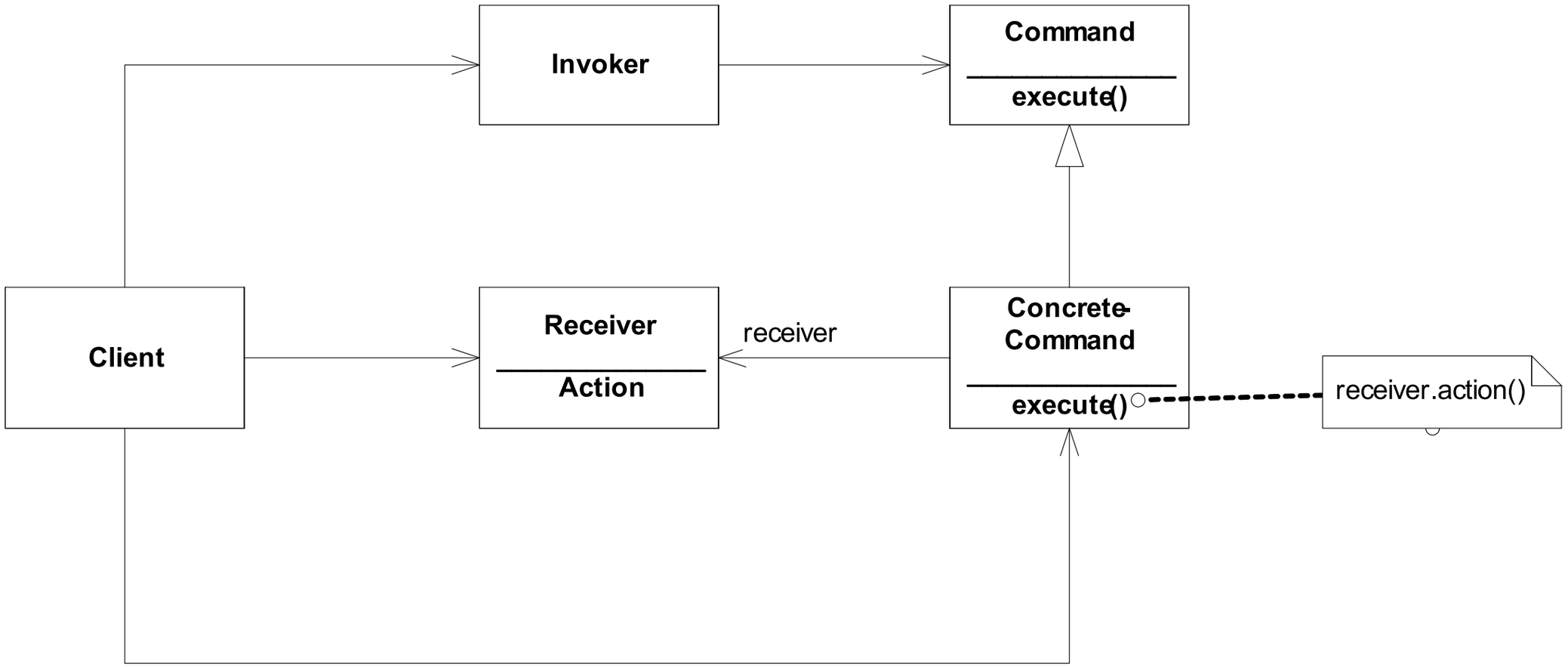
http://elearning.wit.ie

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

Pacemaker Command

Command Pattern Structure



- Encapsulate a request as an object facilitating:
  - parameterize clients with different requests
  - queue or log requests
- + potential support for undoable operations

Command

```
public abstract class Command
{
  protected PacemakerAPI pacemaker;
  protected Parser       parser;

  public Command()
  {}

  public Command(PacemakerAPI pacemaker, Parser parser)
  {
    this.pacemaker = pacemaker;
    this.parser    = parser;
  }

  public abstract void doCommand(Object[] parameters)  throws Exception;
}
```

ListUsersCommand

```
public class ListUsersCommand extends Command
{
  public ListUsersCommand(PacemakerAPI pacemaker, Parser parser)
  {
    super(pacemaker, parser);
  }

  public void doCommand(Object[] parameters) throws Exception
  {
    System.out.println(parser.renderUsers(pacemaker.getUsers()));
  }
}
```

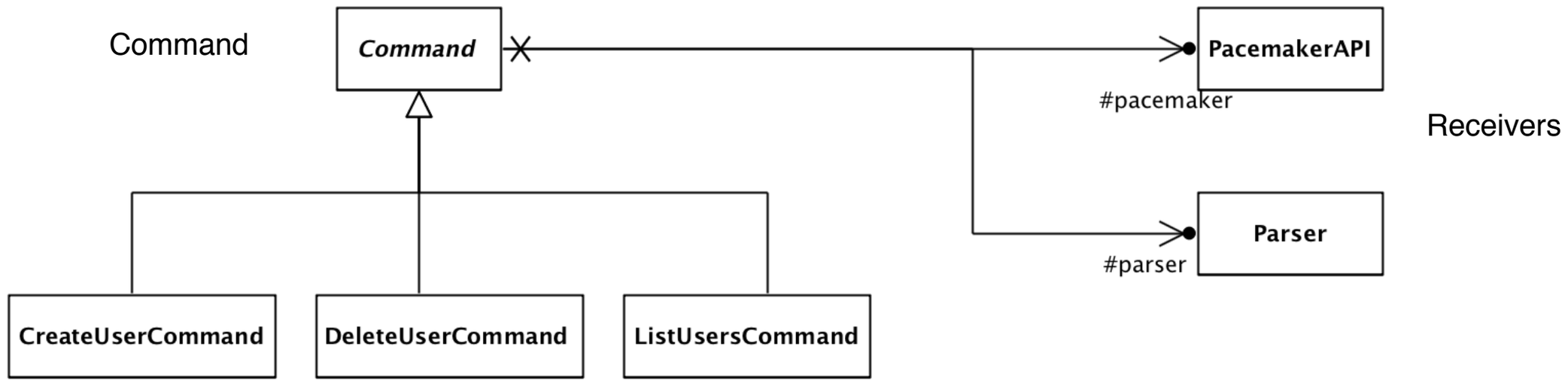
CreateUser Command

```
public class CreateUserCommand extends Command
{
  User user;

  public CreateUserCommand(PacemakerAPI pacemaker, Parser parser)
  {
    super(pacemaker, parser);
  }

  public void doCommand(Object[] parameters) throws Exception
  {
    Long id = pacemaker.createUser((String) parameters[0], (String) parameters[1],
                                   (String) parameters[2],    (String)parameters[3]);
    System.out.println(parser.renderUser(pacemaker.getUser(id)));
    this.user = pacemaker.getUser(id);
  }
}
```

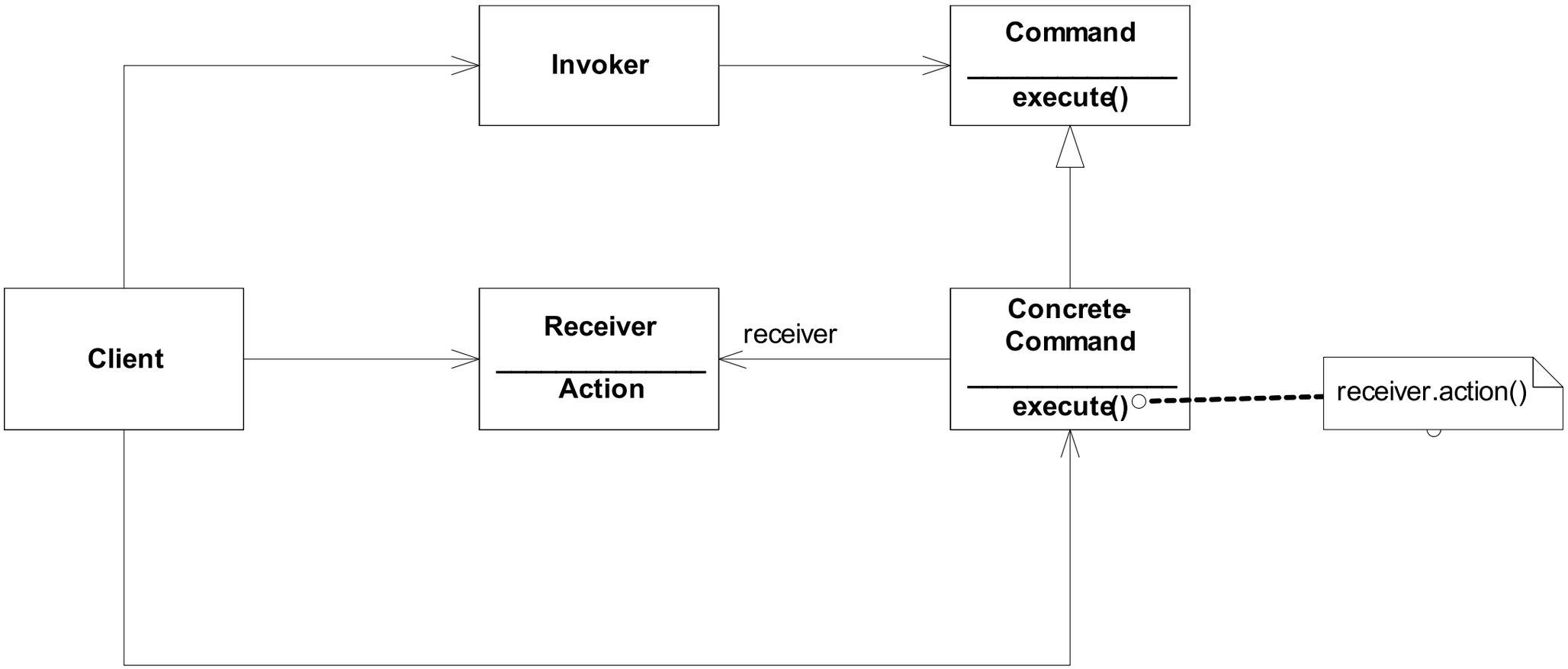
DeleteUserCommand

```
public class DeleteUserCommand extends Command
{
  private User user;

  public DeleteUserCommand(PacemakerAPI pacemaker, Parser parser)
  {
    super(pacemaker, parser);
  }

  public void doCommand(Object[] parameters) throws Exception
  {
    this.user = pacemaker.getUser((Long)parameters[0]);
    pacemaker.deleteUser((Long)parameters[0]);
  }
}
```

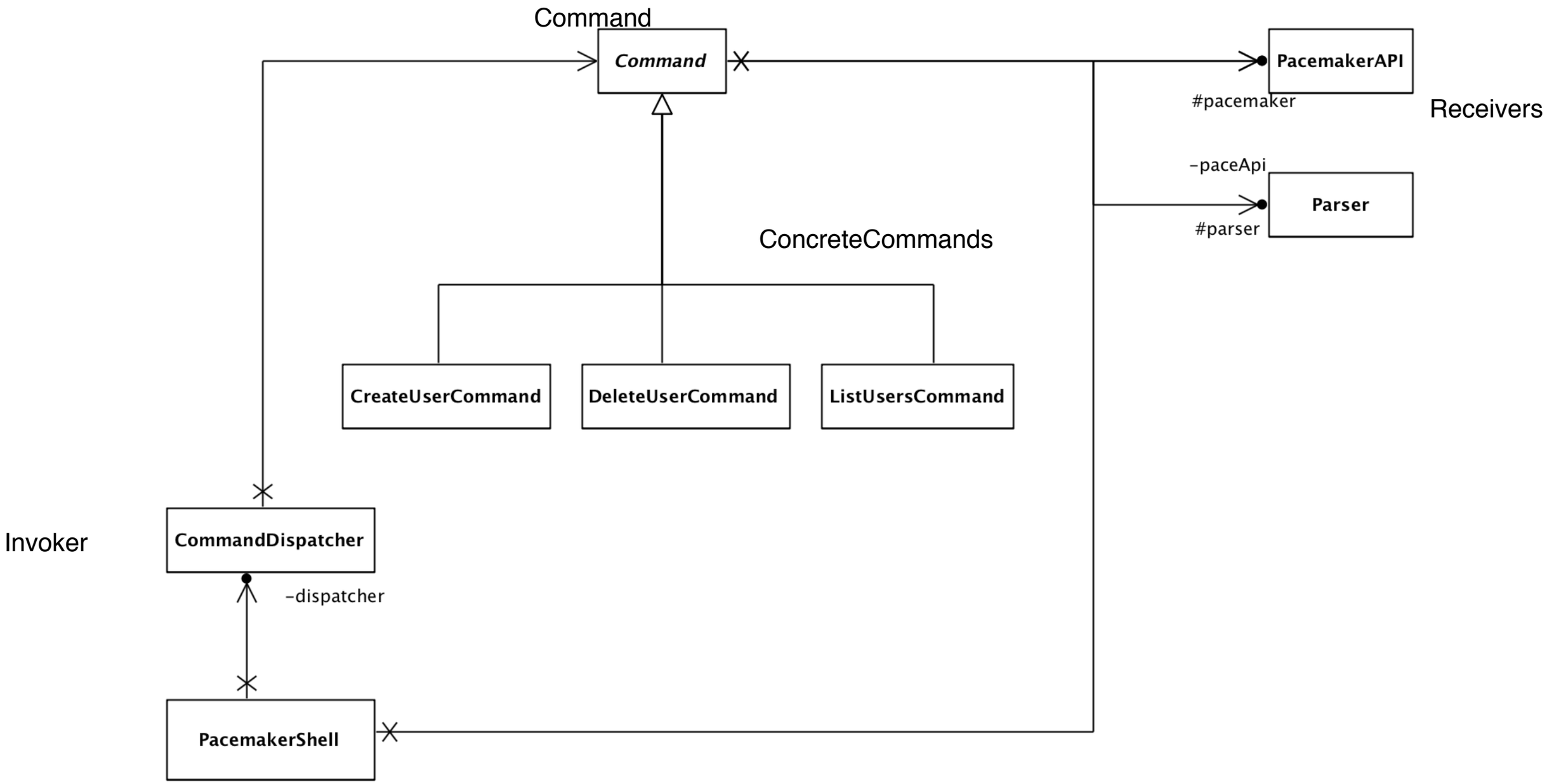
Command Pattern Roles



Invoker?


Invoker
Command
execute()
Client
Receiver
Action
receiver
Concrete-
Command
execute()
receiver.action()

CommandDispatcher



CommandDispatcher

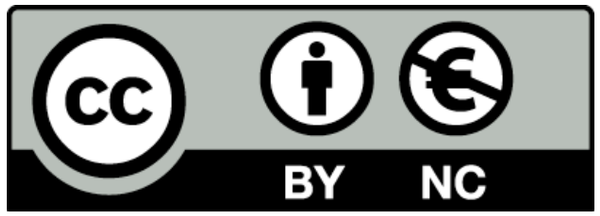
```
public class CommandDispatcher
{
  private Map<String, Command> commands;

  public CommandDispatcher()
  {
    commands = new HashMap<String, Command>();
    commands.put("help", new HelpCommand(commands.keySet()));
  }

  public void addCommand(String commandName, Command command)
  {
    commands.put(commandName, command);
  }

  public boolean dispatchCommand(String commandName, Object [] parameters) throws Exception
  {
    boolean dispatched = false;
    Command command = commands.get(commandName);

    if (command != null)
    {
      dispatched = true;
      command.doCommand(parameters);
    }
    return dispatched;
  }
```

CommandSpecifications

- CommandSpecifications now an empty
- Annotations introspected by Cliche to guide command parsing.
- Dispatching of commands now handled via CommandProcessor strategy

```
public class CommandSpecifications
{
  @Command(description="List all users details")
  public void listUsers () throws Exception
  {}

  @Command(description="undo last command")
  public void undo () throws Exception
  {}

  @Command(description="redo last command")
  public void redo () throws Exception
  {}

  @Command(description="Create a new User")
  public void createUser (@Param(name="first name") String firstname, @Param(name="last name") String lastname,
                          @Param(name="email")      String email,     @Param(name="password")  String password) throws Exception
  {}

  @Command(description="Delete a User")
  public void deleteUser (@Param(name="id") Long id)
  {}

  @Command(description="Help")
  public void help ()
  {}
}
```

```
public class PacemakerShell implements CommandProcessor
{
  private CommandDispatcher dispatcher;
  private PacemakerAPI      paceApi;

  public PacemakerShell()
  {
    Parser parser = new AsciiParser();
    paceApi    = new PacemakerAPI();
    dispatcher = new CommandDispatcher();
    dispatcher.addCommand("list-users",  new ListUsersCommand(paceApi,  parser));
    dispatcher.addCommand("create-user", new CreateUserCommand(paceApi, parser));
    dispatcher.addCommand("delete-user", new DeleteUserCommand(paceApi, parser));
  }

  @Override
  public void doCommand(ShellCommand command, Object[] parameters)
  {
    try
    {
      dispatcher.dispatchCommand(command.getName(), parameters);
    }
    catch (Exception e)
    {
      System.out.println("Error executing command");
    }
  }

  public static void main(String[] args) throws Exception
  {
    PacemakerShell main = new PacemakerShell();
    CommandSpecifications commandSpecs = new CommandSpecifications();

    Shell shell = ShellFactory.createConsoleShell("pm",
                                                  "Welcome to pacemaker-console - ?help for instructions",
                                                   commandSpecs, main);
    shell.commandLoop();
  }
}
```



Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit