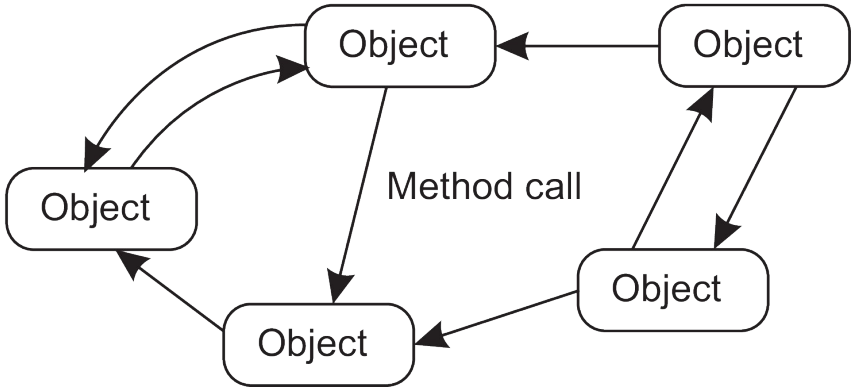
Remote Objects Lab

Object-based architecture

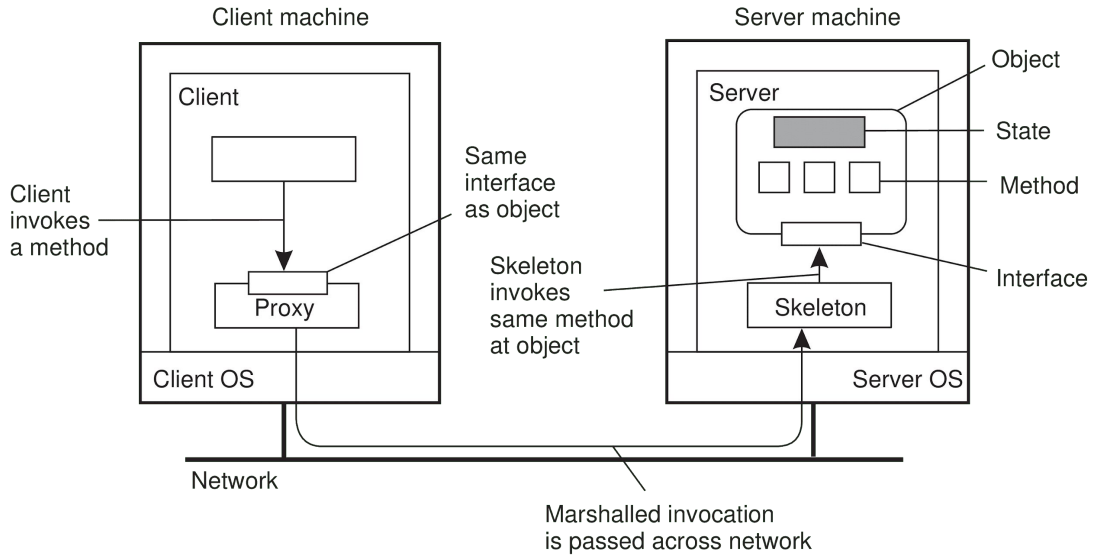
- Objects provide a way of encapsulating data and operations that can be performed on that data.
- Each object corresponds to a software component and these are connected through method calls.



An object-based architectural style.

Remote Method Invocation (RMI)

- RMI is the object-oriented equivalent of (RPC).



Common organisation of a remote object with client-side proxy.

Python Classes and Objects

- Objects provide a way of encapsulating data and operations that can be performed on that data.
- Create a class named MyClass, with a property named x:

```
class MyClass:
    x = 5
```

- Create an object named p1, and print the value of x:

```
p1 = MyClass()
print(p1.x)
```

Object constructor

- All classes have a function called __init__(), which is always executed when the class is being initiated.

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

p1 = Person("John", 36)
print(p1.name)
print(p1.age)
```

- The self parameter is a reference to the class itself.

Object Methods

- Objects can also contain methods; functions that belongs to the object.

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def myfunc(self):
        print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
```

Name Server

- Network service for mapping between the names of resources in a distributed system and their respective network addresses.
- Often translates a humanly meaningful name to a network address.
- An example of a name server is the server component of the Domain Name System (DNS).

Pyro - Python Remote Objects

- Python library for remote objects; this is a type of middleware (https://pyro4.readthedocs.io/en/stable/)
- In lab you will implement a basic database system using Pyro.
- Install Pyro using Pipenv

Using Pipenv

- Open a command line terminal, and lets create a directory where we will keep all the code for our project:

```
$ mkdir pipenv-pyro
$ cd pipenv-pyro
```

- Now we are in the directory for our project, we can use pipenv to create a virtual environment and install our first library.

```
$ pipenv install Pyro4
```

- Copy python files for lab to the pipenv-pyro directory.

- Now the libraries are installed we can use them in some Python code.
- To run the name server enter

```
$ pipenv run python -m Pyro4.naming
```

- To run the server enter

```
$ pipenv run python warehouse.py
```

- To run the server client

```
$ pipenv run python visit.py
```