2.1.3.2 Object Oriented Programming

Return to Programming Paradigm

Object-Oriented Programming (OOP) is a form of Imperative and procedural programming. The main difference is how the procedures, functions and data are organized. OOP encapsulates data and behavior into objects. In other words, an object is the definition of data and the operations that can be performed on that data. OOP applications use a collection of objects to accomplish the overall goal of the program or application. For example an object represent a person (i.e., a person-object). The person-object has, by definition, certain attributes or properties which are made public or held private. In addition, the person-object has methods that ensure the validity of the name attribute within the person-object as defined by the person-object. OOP also has abstraction for objects called classes that are blueprints for real world instances of things. In the example above, there is class of objects called people. Each individual person is represented by an instance of the people-object (i.e., john-people-object, and jane-people-object). All the instances of the people-objects offer the same attributes and methods.

Some examples of Object Oriented Programming languages are:

- C++
- Java
- JavaScript
- Python
- Ruby
- PHP
- Scala
- Visual Basic .NET

In Figure 1, the green boxes represent functions (i.e., operations) and the blue cylinders represent data associated with the an Object represented by the outer orange boxes. Objects encapsulate the data objects and the operations that work on that data. The data flow and control is accomplished by invocation of operations (i.e., methods) between the different objects.

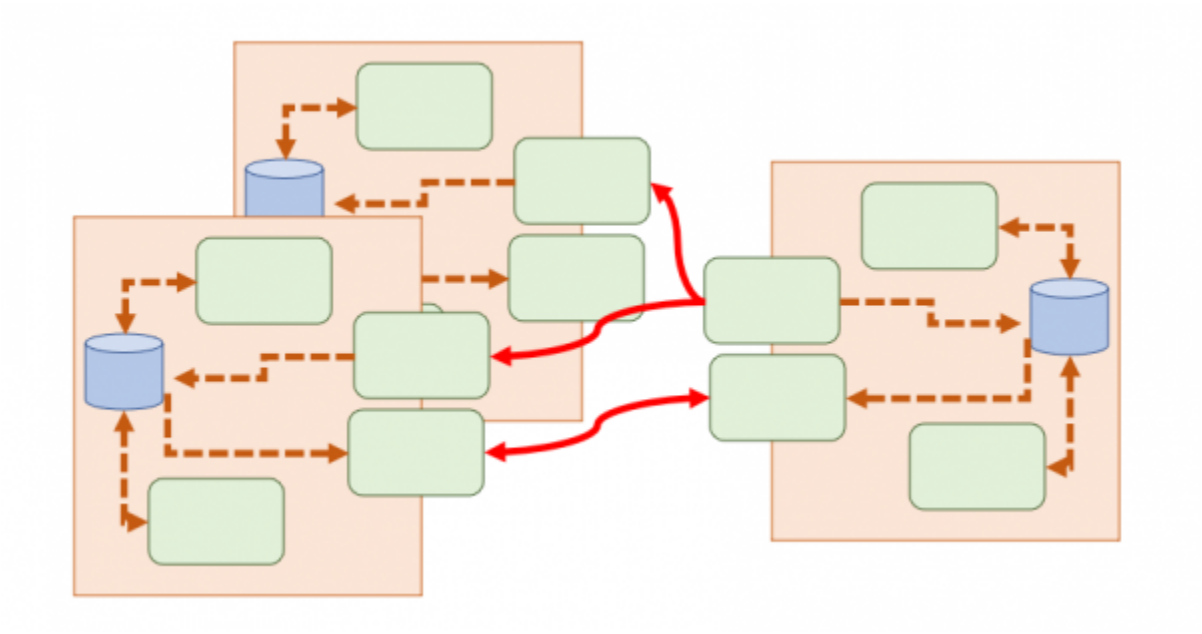

Figure 1: Object-Oriented (OO) Data Flow