In the literature we can find a plethora of libraries of optimization algorithms implemented in different programming languages (for example, evolutionary algorithm libraries); all with their own advantages and drawbacks. However, these libraries can only be used by people knowing the programming language in which the library is implemented and, even in this way, some time is required to be familiar with the algorithms and work with them. Our proposal consists of providing a service for the remote execution of optimization algorithms through Internet, ROS, Remote Optimization Service [1]. This way, different algorithms developed in different programming languages are grouped in a repository reachable from anywhere in the world through Internet. Two types of actors that interact with the system are established: the developer, who, by means of a the Wrapper design pattern, adds an algorithm in a server machine (Worker), and the user, who, using the client application, can interact with the available algorithms in a remote way for solving optimization problems. Moreover, it is possible to configure ROS for working in a single machine as well as distributed in a network of workstations.

The Primary Server is in charge of the user authentication. It also offers information of the algorithms and servers available to the user. The Distribution Server (Server in Figure 1) is a bridge between connected clients and process servers. The distribution server is therefore a fundamental piece to obtain scalability, flexibility, and the system distribution in network, since it hides the true location of process servers and controls the load balancing among them. Finally, the Worker houses the algorithms, and executes them according to the orders specified by the client and routed through the distribution server. Once an execution is finished, it sends the results to the distribution server, which forwards them to the client. Let us illustrate a normal use case of ROS to get an idea of the data flow sequence through the system in a normal operation. We follow the numeration and direction of dotted lines in Figure 1.

1. **User authentication**. The user fills the initial form of the ROS client with his identification and password and connects to the primary server.

2. **Selection of algorithm, server, and programming language**. The primary server sends a list of algorithms available, distribution server addresses, and programming languages. The client receives this information and shows it to the user, who makes a selection.

3. **Execution configuration**. The client, based on the algorithm, server, and programming language selected in the last step, asks to the distribution server the specific parameters and configuration options (via XML) that the user must fill in order to run the algorithm.

4. **Algorithm execution**. Once the user sets the algorithm parameters, the XML file generated filled by the client is sent to the distribution server. This server sends the XML file to the suitable process server, which starts the execution of the algorithm.

5. **Results collection**. After the end of the algorithm execution, the results are sent in XML format to the distribution server, which forwards them to the client.

The installation package of ROS is free available at the URL http://tracer.lcc.uma.es/ros/index.html. It is possible to evaluate the system connecting it to active servers already working in our institution and whose IP addresses are configured in the client application.

1. **REFERENCES**