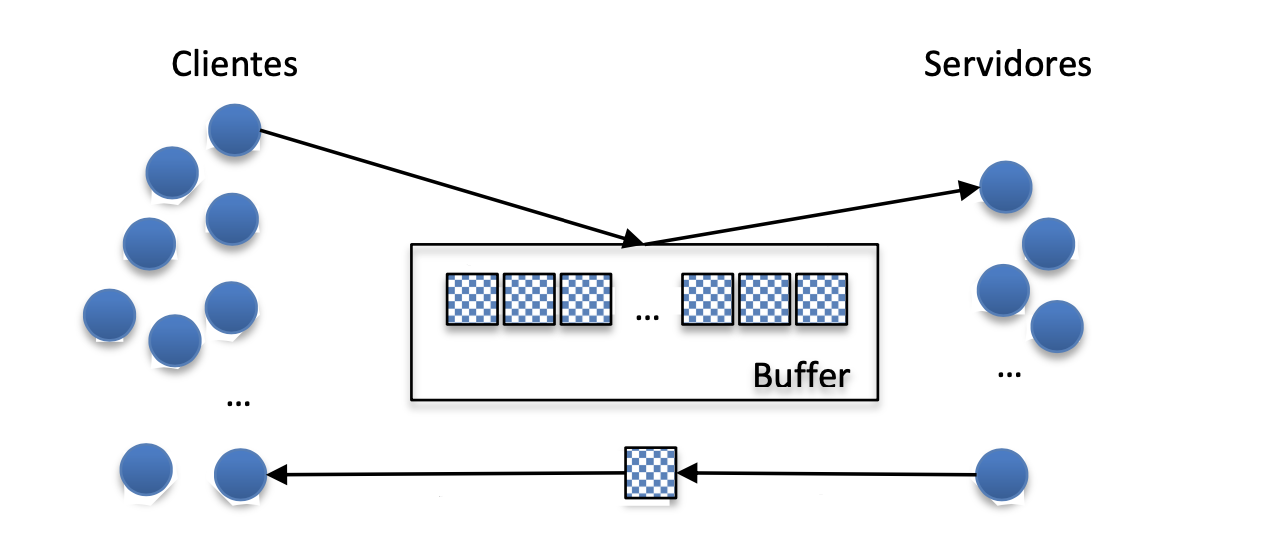
ISIS 2203 Caso 1

Concurrency Management

Esteban Cardenas - 201813011  
Ryan Bosher - LT716571



# Program Explanation

**Introduction**

In Caso 1 we were expected to solve numerous technical challenges by utilizing Java concurrency. The system designed had to be flexible to account for the variable parameters and robust enough to handle any combination. For our project submission we designed and tested a system created to adhere to the supplied brief and this document will explain the reasoning for our implementation.

**Program design**

The design of our solution follows the pattern of a Producer Consumer model utilizing a buffer. Clients produce messages which are loaded into the buffer, then in turn retrieved by servers from the buffer. Clients and servers do not interact directly, however indirectly through the buffer. When the clients are created, a message queue is created using the **Queue.enqueue()** method, with one queue being assigned per client. The queue is filled using nested for loops to create the correct number of messages per client. Servers are created in a similar manner and all **runnable()** objects are started and have the **join()** method called at the end of the **main()** method. This order of operations enables synchronization among clients and servers.

Once clients have created and sent messages, they enter a state of active waiting while servers read the messages and reply to the buffer. Once a reply has been sent to the buffer, the buffer uses the **notify()** method to alert the waiting client that a reply has been received. The message is read and discarded. This process repeats until there are no messages left in the queue, at which point the clients **yield()** then servers **yield()**. With no active threads running, the program terminates successfully.

**Object interactions**

Following the program design, the primary interaction between objects in our system is between Clients and the Buffer and Servers and the Buffer. All 3 primary classes also interact with Message objects in different capacities. Clients create and send messages to the buffer and read the response. Buffers facilitate message delivery to both Clients and Servers. Servers read client messages and send responses to the buffer. Messages are not usually directly accessed, but by way of the Queue object. This class was imported from <https://algs4.cs.princeton.edu/> and is used in accordance with the license conditions.