

---

---

# TRABAJO FINAL DE SISTEMAS OPERATIVOS

---

---

*Bruno Baruffaldi*  
*Mauricio Muratori*

*Universidad Nacional de Rosario*  
*Facultad de Ciencias Exactas, Ingeniería y Agrimensura*



## 1. Introducción:

A continuación se muestran los detalles de la implementación del trabajo final de Sistemas Operativos I. El mismo consiste en un servidor de Ta-Te-Ti implementado en el lenguaje Erlang. El código esta separado en distintos módulos:

- server.erl
- dispatcher.erl
- psocket.erl
- pstat.erl
- pcomando.erl
- pbalance.erl

Se presenta ademas un archivo client.erl que ejemplifica un cliente básico para comunicarse con el servidor.

## 2. Desiciones de Diseño:

### 2.1. Comunicación:

El servidor por defecto espera nuevas conexiones a través del puerto 8000 e interactuá con los clientes utilizando sockets TCP.

Para el cliente el servidor funciona como un único programa que se ejecuta en una sola maquina, aunque este puede estar compuesto por distintos nodos que se comunican entre si aprovechando las ventajas del lenguaje Erlang.

La forma general de los comandos que el servidor podrá responder es CMD comdid op1 op2 ..., donde CMD es una de los comandos disponibles, comdid es un identificar unico del comando.

La respuesta del servidor sera de la forma OK comdid op1 ... o ERROR comdid op1 ..., aunque ademas el servidor podrá enviar al cliente mensajes de la forma UPD op1 op2 ...

### 2.2. Arquitectura:

- Server: Se encarga de la sincronizacion de los distintos nodos del sistemas y maneja la información de los jugadores y las partidas.
- Dispatcher: Su tarea es esperar y atender nuevas conexiones.

- Psocket: Una vez establecida la conexión, se encarga de la comunicación entre el cliente y el servidor a través del socket TCP. Implementa además un pequeño parser de comandos.
- Pstat: Su función es evaluar la carga del nodo e informar acerca de esta a intervalos regulares.
- Pcomando: Atiende las consultas del cliente, comunicándose con los distintos módulos.
- Pbalance: Se encarga de decidir que nodo del sistema es el más apropiado para atender la consulta del cliente.

### 3. Funcionamiento:

Cada nodo se ejecutará en una máquina virtual de Erlang, por lo que es necesario tenerlo instalado en todas las máquinas donde el servidor se funcione.

Para utilizar el servidor se debe ejecutar los siguientes comandos:

```
erl -name NameNode@IP -setcookie Cookie
```

Una vez hecho esto es necesario conectar los nodos entre sí:

```
net_adm:ping('nodeName@IP').
```

Donde 'nodeName@IP' es uno de los nodos del servidor. Finalmente, se ejecuta

```
server:init().
```

y el servidor empezará a funcionar.

Por parte del cliente, en una máquina virtual de Erlang escribimos

```
client:init().
```

y ya tendremos el cliente conectado al servidor.