Comunicación

Laboratorio remoto: Viga simplemente apoyada

UNIVERSIDAD DE LA MARINA MERCANTE, CABA, BUENOS AIRES ARGENTINA

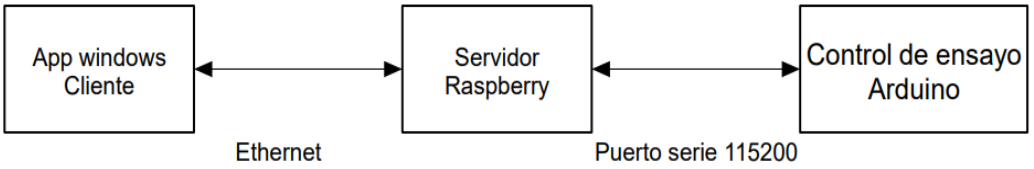
leandro.cintioli@alumnos.udemm.edu.ar

luis.villacorta@alumnos.udemm.edu.ar

pablo.tavolaro@alumnos.udemm.edu.ar

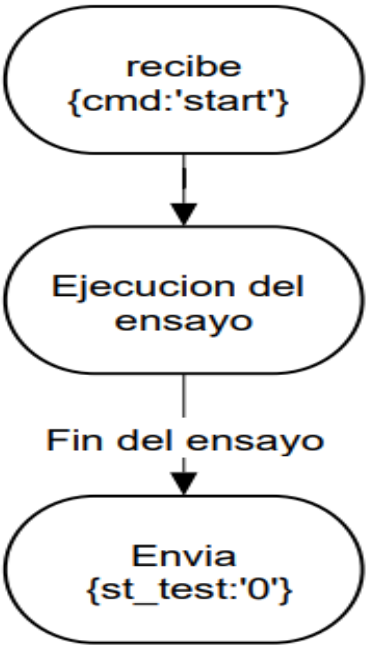
Profesor Tutor: Marcelo Bellotti

Comunicación



Arduino mega, banco de ensayo.

El Arduino se encarga de ejecutar el ensayo. Por medio del puerto serie se le pueden enviar y recibir json. Al recibir el comando de start , ejecuta el ensayo, cuando termina devuelve el status de test terminado.



Antes de comenzar el ensayo se deben setear los parámetros del ensayo, esto se logra mandando el formato del json a modificar

{distance:'500'} distance:0 a 254 Distancia en mm donde se aplica la fuerza.
{force:'11'} force:0 a 254 Fuerza a aplicar en Kg. Cuando el ensayo no se esta ejecutando el arduino , se le piden los valores de los resultados del ensayo.

{reaction_one:'1'} reaction_one : uint_32 Fuerza de reacción uno, en g.
{reaction_two:'2'} reaction_two :uint_32 Fuerza de reacción dos, en g.
{flexion:'3'} flexion : uint_32 Flexión del ensayo, en cm.

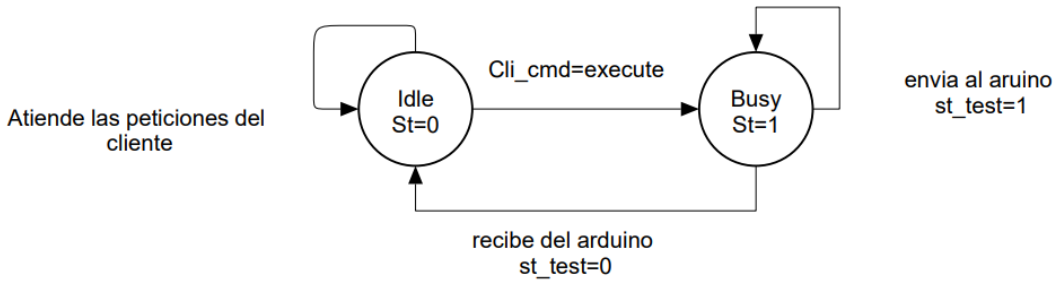
Esto se hace mediante :
{info:'reaction_one'} Devuelve la reaction1 del ensayo.
{info:'reaction_two'} Devuelve la reaction2 del ensayo.
{info:'flexion'} Devuelve la flexión del ensayo.

Raspberry Pi 3 ,servidor

El servidor recibe los pedidos del cliente.

Tiene dos estados ,

Busy: realizando el experimento
Idle: respondiendo peticiones del cliente.



El servidor código en python del servidor se encuentra en:

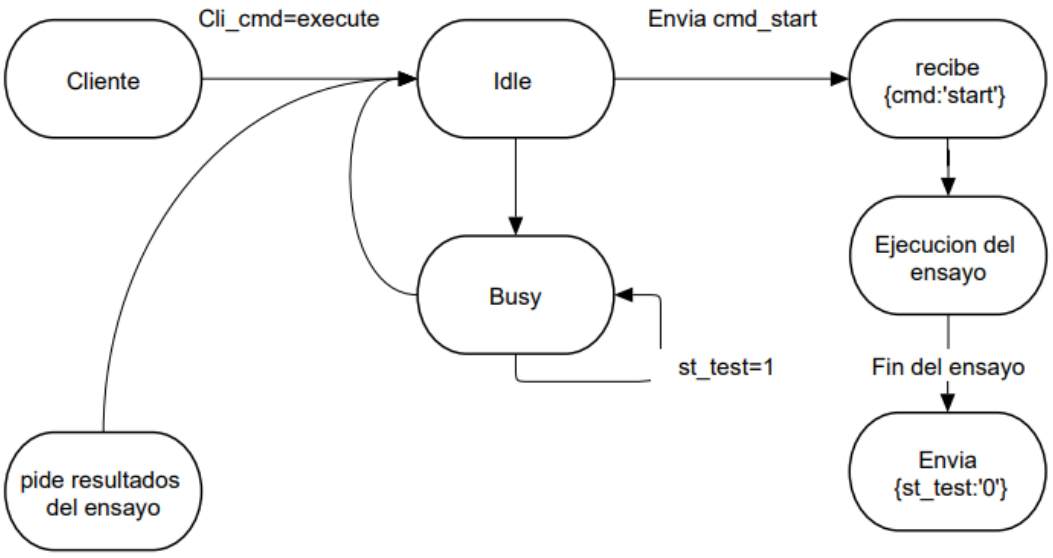
https://github.com/theinsideshine/beam_remote_lab_server

Para Correr el servidor
ejecutar en rasperry python beam_remote_lab_server.py

Insomnia-Postman Cliente

Para simular al cliente se usó insomnia recordar que estos tipos de clientes tiene integrado los credenciales de seguridad para CORS

Se establece que los datos en el servidor son del último experimento realizado. El cliente cuando recibe el status de Idle del servidor, interpreta que está en condiciones de leer los parámetros del último ensayo realizado.



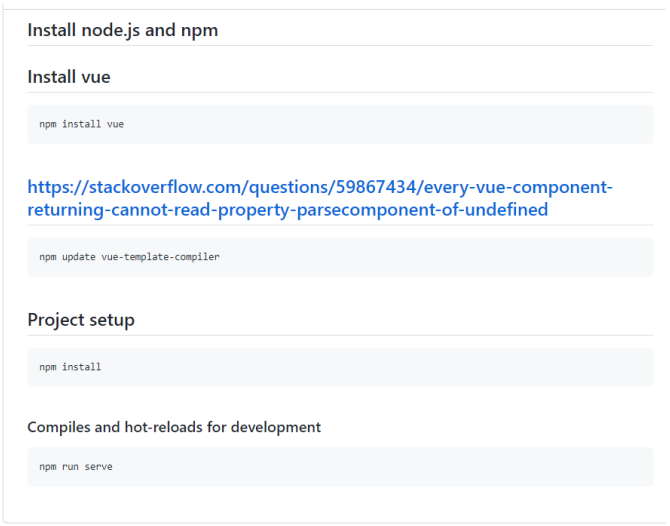
Cliente

Se escribió un cliente en el entorno de desarrollo VUe
La intención inicial es poder leer los parámetros del arduino y controlar el ensayo.
Se subió a las páginas de github el deploy del front end

.https://theinsideshine.github.io/frontend-beam/

el código se encuentra en :

https://github.com/theinsideshine/frontend-beam/tree/master



Install node.js and npm

Install vue

```
npm install vue
```

https://stackoverflow.com/questions/59867434/every-vue-component-returning-cannot-read-property-parsecomponent-of-undefined

```
npm update vue-template-compiler
```

Project setup

```
npm install
```

Compiles and hot-reloads for development

```
npm run serve
```

La interfaz cliente servidor es la siguiente
Las peticiones del cliente son:

```
@app.route('/ping')
@app.route('/info/parameters') #pide todos los parámetros del ensayo
@app.route('/info/calibration') #pide todo los parámetro de calibración
@app.route('/info/version')#pide la versión del firmware
@app.route('/info/status') #pide el status del ensayo st_test
@app.route('/info/reaction_one') #pide la fuerza de reacción 1
@app.route('/info/reaction_two') #pide la fuerza de reacción 2
@app.route('/info/flexion') #pide la distancia de flexión
@app.route('/parameters/distance/<string:distance>', methods=['PUT'])#Modifica la distancia del ensayo
@app.route('/parameters/force/<string:force>', methods=['PUT']) #Modifica la fuerza del ensayo
@app.route('/parameters/step_k/<string:step_k>' #Modifica la constante de flexión
@app.route('/comands/start', methods=['PUT']) #Comienza el ensayo
```