

Nombre: Jorge Alejandro Solano Sandoval

Materia: Cinemática de robots

Carrera: Ing. Mecatronica

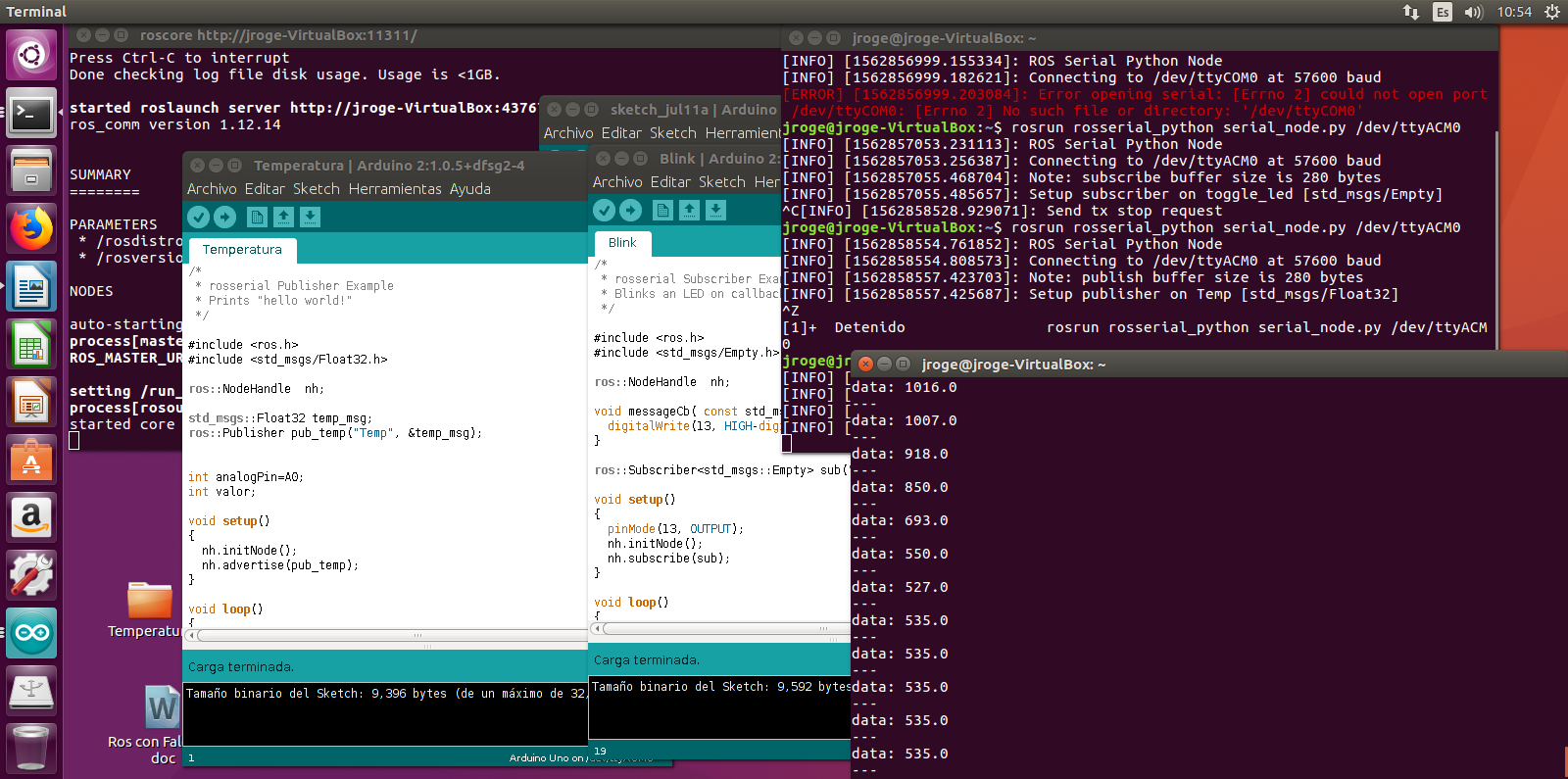
Maestro: Carlos Moran Garabito

Sistema Operativo Robótico (en inglés Robot Operating System, ROS) es un [framework](https://es.wikipedia.org/wiki/Framework) para el desarrollo de software para [robots](https://es.wikipedia.org/wiki/Robot) que provee la funcionalidad de un [sistema operativo](https://es.wikipedia.org/wiki/Sistema_operativo) en un [clúster](https://es.wikipedia.org/wiki/Cluster_(inform%C3%A1tica)) heterogéneo. ROS se desarrolló originalmente en 2007 bajo el nombre de *switchyard* por el [Laboratorio de Inteligencia Artificial de Stanford](https://es.wikipedia.org/w/index.php?title=Laboratorio_de_Inteligencia_Artificial_de_Stanford&action=edit&redlink=1) para dar soporte al proyecto del Robot con [Inteligencia Artificial](https://es.wikipedia.org/wiki/Inteligencia_Artificial) de Stanford ([STAIR](http://stair.stanford.edu/)[2](https://es.wikipedia.org/wiki/Sistema_Operativo_Rob%C3%B3tico#cite_note-2)​). Desde 2008, el desarrollo continúa primordialmente en [Willow Garage](https://es.wikipedia.org/wiki/Willow_garage), un instituto de investigación robótico con más de veinte instituciones colaborando en un modelo de desarrollo federado.

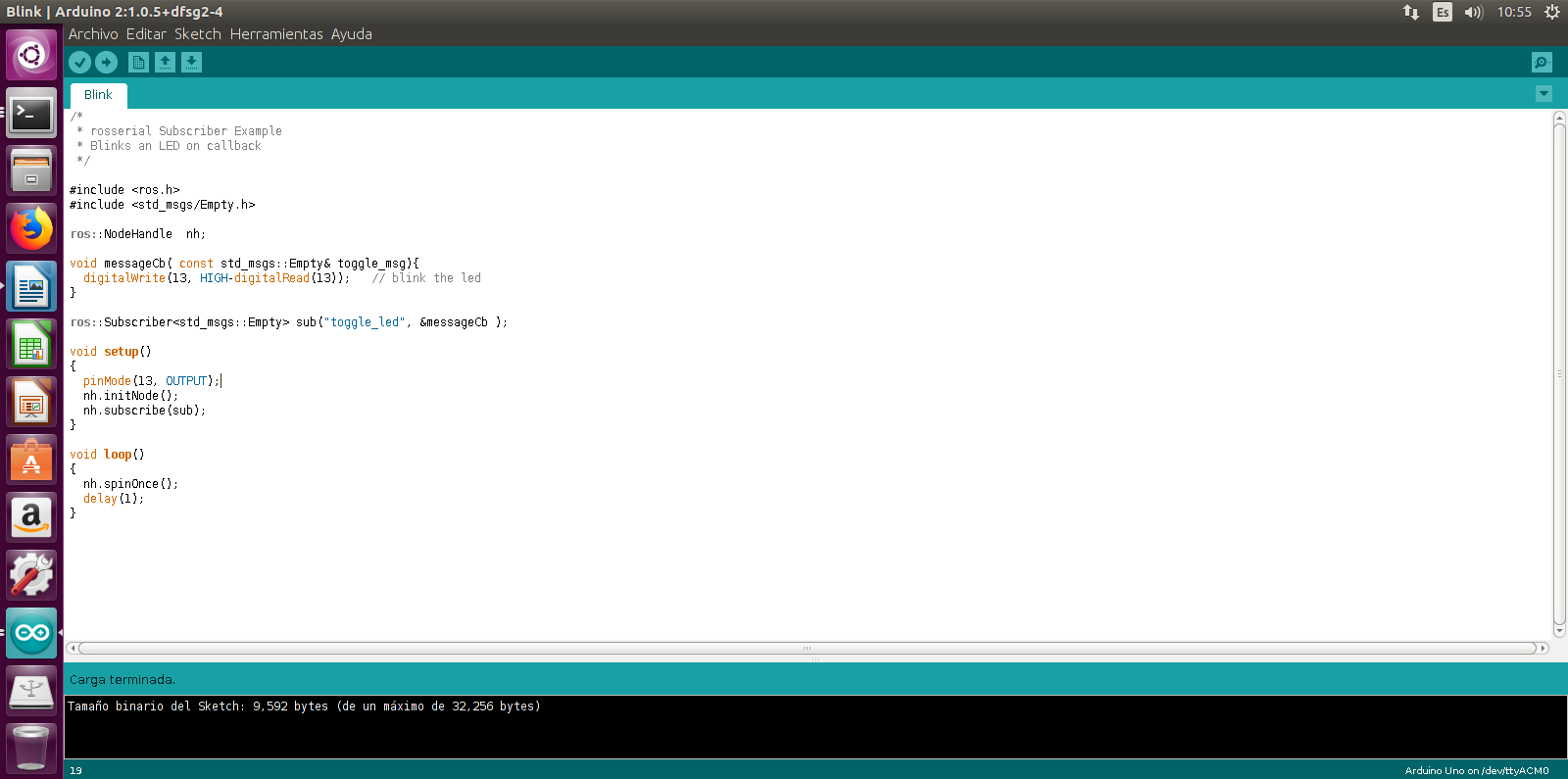
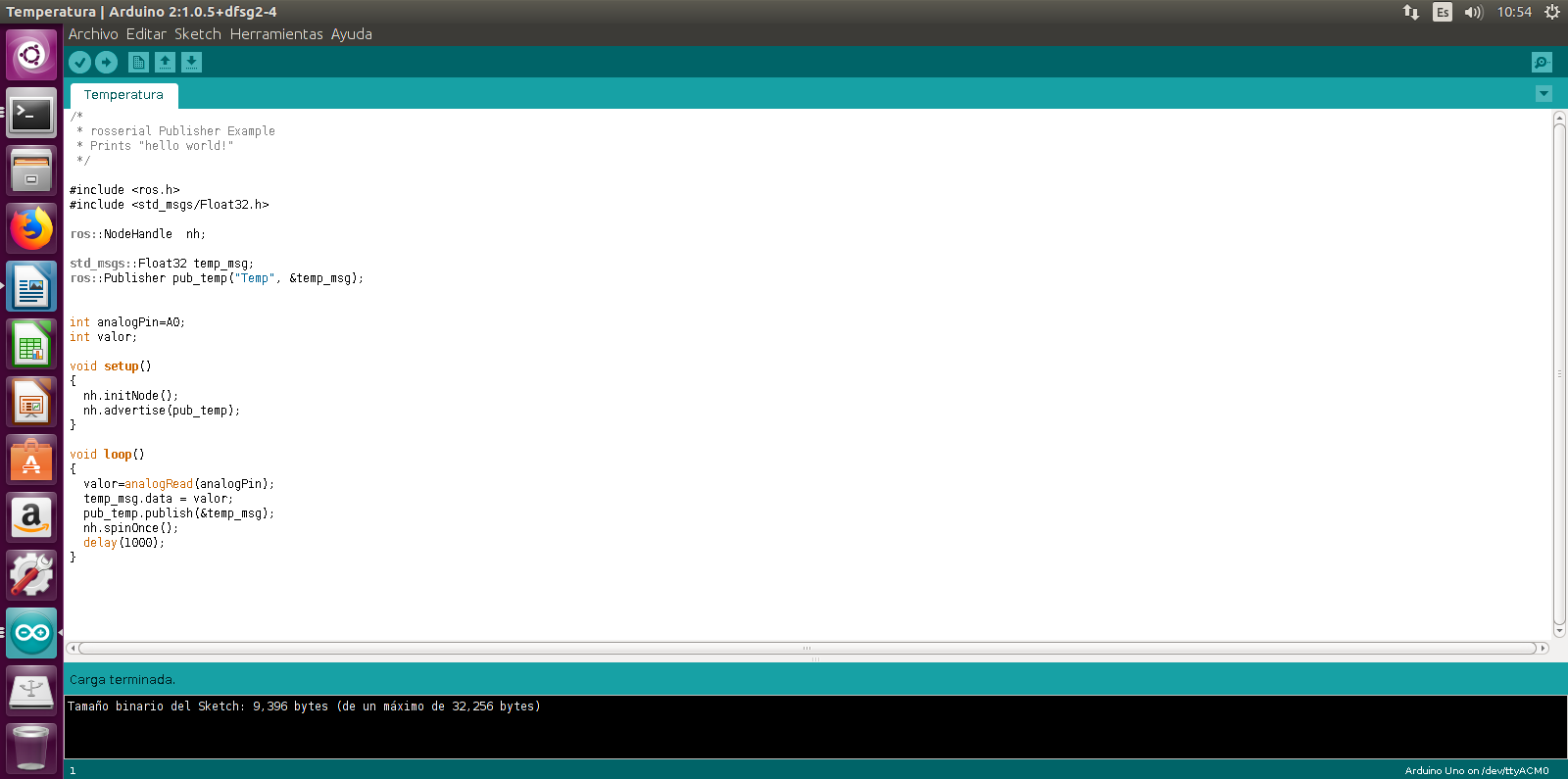
ROS provee los servicios estándar de un sistema operativo tales como abstracción del hardware, control de dispositivos de bajo nivel, implementación de funcionalidad de uso común, paso de mensajes entre procesos y mantenimiento de paquetes. Está basado en una arquitectura de [grafos](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_grafos) donde el procesamiento toma lugar en los nodos que pueden recibir, mandar y multiplexar mensajes de sensores, control, estados, planificaciones y actuadores, entre otros. La librería está orientada para un [sistema UNIX](https://es.wikipedia.org/wiki/Unix-like)([Ubuntu](https://es.wikipedia.org/wiki/Ubuntu) (Linux)) aunque también se está adaptando a otros sistemas operativos como [Fedora](https://es.wikipedia.org/wiki/Fedora_(distribuci%C3%B3n_Linux)), [Mac OS X](https://es.wikipedia.org/wiki/Mac_OS_X), Arch, [Gentoo](https://es.wikipedia.org/wiki/Gentoo), [OpenSUSE](https://es.wikipedia.org/wiki/OpenSUSE), Slackware, [Debian](https://es.wikipedia.org/wiki/Debian) o [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows), considerados como 'experimentales'.

ROS tiene dos partes básicas: la parte del sistema operativo, *ros*, como se ha descrito anteriormente y *ros-pkg*, una suite de paquetes aportados por la contribución de usuarios (organizados en conjuntos llamados *pilas* o en inglés *stacks*) que implementan la funcionalidades tales como localización y mapeo simultáneo, planificación, percepción, simulación, etc.

ROS es [software libre](https://es.wikipedia.org/wiki/Software_libre) bajo términos de [licencia BSD](https://es.wikipedia.org/wiki/Licencia_BSD). Esta licencia permite libertad para uso comercial e investigador. Las contribuciones de los paquetes en *ros-pkg* están bajo una gran variedad de licencias diferentes.

En la imagen que se encuentra a continuación se muestran los valores que arroja el sensor que en este caso es un potenciómetro, para hacer esta actividad primero se corre roscore, después se realiza una búsqueda del dispositivo externo con un comando que es tty después que pones ese comando te va a arrojar todos los dispositivos que están conectados al computador, después de eso ubicas el dispositivo que vas a comunicar con ROS para realizar la lectura, una vez localizado introducir comando que en este caso es rorun rosserial\_python serial\_node.py /dev/ttyACM que en este caso la última función es acorde al puerto en donde tengo conectada la tarjeta.

Después de esto se pone el comando del check list en donde me va a desplegar la funciones que puedo utilizar de acuerdo a mi programa en este caso fue temp que fue como tenía el nombre del programa

En la siguiente imagen se muestra el programa que se compilo al arduino esto para lograr tener lectura del sensor, se declaran dos variables una de entrada y otra para almacenar la variable leída.

CONCLUSION

En esta tarea lo más complicado que se me hizo fue poder hacer la comunicación con ros y la tarjeta en este caso arduino y cypres, las dos tarjetas fue muy complicado realizar la comunicación ya que yo estoy trabajando con máquina virtual y mi computadora por ello no tiene tanta capacidad para poder correr la máquina virtual y en ocasiones se volvía lenta la simulación por ello me costaba más problema darme cuenta de los errores de instalación, ya que al momento de instalar ros y todos los paquetes correspondientes se tienen que instalar sin ningún problema porque si no el programa que quiero correr no va a poder ni siquiera arrancar, tuve que instalar roscore varias veces porque el comando que usaba estaba erróneo por eso no me dejaba correr el programa.

El programa en cypres me costó trabajo realizarlo ya que tenía mucho tiempo que no utilizaba esa aplicación pero se logró y pude realizar la tarea.