Entorno de Contenedores de Emuladores que contienen Sistemas Embebidos

Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca, Raúl Villca, Matías Adagio
Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza
Dirección: Florencio Varela 1703 – CP 1754 – {wvaliente, ecarnuccio, mvolker, gdeluca}@unlam.edu.ar, {raul.villcasd, mati.adagio}@gmail.com

RESUMEN

Aunque la población no se percate, cada vez más, los sistemas embebidos forman parte de la vida cotidiana de las personas. Como consecuencia de dicho auge de la tecnología, surgieron en el mercado diferentes sistemas embebidos conformados por placas de desarrollo, que inicialmente se utilizan para prototipado. Estas son de fácil aprendizaje, lo que permitió que sean muy utilizadas en los últimos años. Por ese motivo para un usuario sin experiencia, resulta un desafío la elección de la placa correcta para sus proyectos. Esto se debe a que muchas veces se puede equivocar en dicha selección, ya que no cumpliría con sus expectativas. Por ende, en esta investigación se pretende minimizar esos riesgos, a través de la utilización de emuladores que permitan imitar el comportamiento de estos sistemas embebidos. Aprovechando estas características se procura generar un entorno de integración, empaquetado y automatizado mediante contenedores. Los cuales permitirán al usuario probar determinados programas rápidamente, en el hardware emulado del sistema embebido, sin necesidad de incurrir en costos. De forma tal de permitirle determinar si la placa elegida puede llegar a cumplir sus expectativas, sin necesitar adquirir el hardware físico.

Palabras clave: Sistemas Embebidos, Emuladores, Qemu, Contenedores, Docker.

CONTEXTO

Nuestra Línea de Investigación es parte del proyecto "*Entorno de integración continua para validación de sistemas embebidos de tiempo real"*, dependiente de la Unidad Académica del *Departamento de Ingeniería e Investigaciones Tecnológicas*, perteneciente al programa de Investigaciones CyTMA2 de la Universidad Nacional de La Matanza, el cual es formado por docentes e investigadores de la carrera de ingeniería en informática. Este proyecto es continuación de los trabajos que viene realizando el grupo de investigación en sistemas operativos, computación de alto rendimiento y en el área de Internet de las cosas.

1. INTRODUCCIÓN

En los últimos años los sistemas embebidos han tenido un crecimiento exponencial, debido a que son utilizados en la vida cotidiana de las personas con mayor frecuencia. Este gran auge se debe principalmente a sus características que son su miniaturización, su menor consumo

 RedUNCI - UNdeC. ISBN: 978-987-24611-3-3

energético, su gran capacidad de procesamiento y por la interconexión entre ellos. Gracias a su evolución, han surgido nuevos conceptos tecnológicos, los cuales hace décadas atrás eran impensados. Tales como dispositivos vestibles, domótica, telefonía móvil, internet de las cosas, ciudades inteligentes, industria 4.0, telemedicina, entre otros. A su vez algunos de ellos se pueden combinar con otras nuevas herramientas de procesamientos de datos, tales como Big Data o Machine Learning. Esto se debe a que existen sistemas embebidos que recompilan gran cantidad de información, para obtener conocimiento de su entorno y así aprender de los datos en forma automática.

Durante la evolución de los sistemas embebidos, hubo un punto importante en su línea de tiempo. Massimo Banzi en el año 2005 inicia, junto a sus alumnos, el proyecto Arduino en el instituto IVRAE en Italia. Este tenía como premisa permitirles aprender a trabajar, en muy poco tiempo y de forma económica, con microcontroladores a sus estudiantes. El proyecto se fundamentó en el diseño, construcción y prototipado de hardware libre. Además de ser multiplataforma y de fácil aprendizaje a través del lenguaje Wiring. [1] Estas características permitieron que en poco tiempo las placas Arduino tuvieran una gran aceptación, de forma tal que actualmente existe una gran comunidad de usuarios que aportan sus conocimientos y constantemente van optimizando el proyecto. [2]

A partir del surgimiento de las placas de prototipado Arduino, emergieron otros sistemas embebidos que tomaron como base las premisas del proyecto de Massimo Banzi, ya que permiten el rápido aprendizaje de su uso e intentan ser de costo accesible para los usuarios. Actualmente existen una gran cantidad de familias de placas de desarrollo en el mercado que pretenden seguir dicho principio. De esta manera, junto con Arduino, las placas más importantes y buscadas por los usuarios para sus proyectos son las familias de Raspberry Pi, las familias de placas que poseen los microcontroladores STM32, las ESP8266 y su sucesor, que poseen el microprocesador ESP32. Siendo Arduino la más buscada entre todas las demás. Esta preponderancia se puede visualizar en el gráfico siguiente [3].

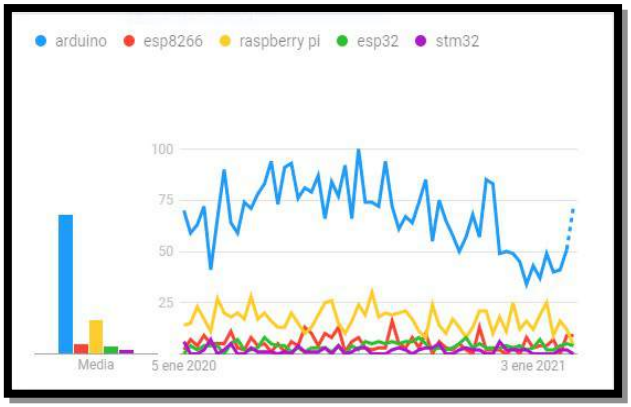

Fig. 1 interés a lo largo del tiempo de los usuarios de distintas placas de desarrollo en Argentina

En Argentina las placas de la familia Arduino es la más buscada y consultada en Internet. Mientras que el resto se mantienen muy por debajo de estas, en cantidad de búsquedas. Diferente es la situación en los países de América del Norte y Europa, en donde la cantidad de búsquedas están lideradas no solo por Arduino, sino también por la Raspberry Pi. Mientras que el resto se mantienen muy por debajo de ellas. Esta tendencia se debe a diversos factores. Por un lado, los STM32 son más potentes que las Arduino, pero a veces no presentan bibliotecas drivers desarrolladas para determinados dispositivos. Mientras que, para Arduino, los drivers son compatibles con la gran mayoría de los periféricos. A su vez los microcontroladores de Arduino, pueden continuar funcionando hasta que las baterías se agoten por completo. En cambio, los STM32 funcionarán hasta alcanzar el umbral de carga

de energía definido por el fabricante [4]. Por su parte las Raspberry Pi, debido a su gran capacidad de procesamiento, posibilidad de ejecución de distintos Sistemas Operativos de escritorio y de conexión de periféricos y conectividad al exterior, se utilizan principalmente como Servidores, Gateway o central de procesamiento de datos e imágenes. No obstante, debido a su alto costo no suelen ser utilizadas para la lectura de sensores y la activación de actuadores, ya que se emplean como intermediarios. Por otro lado, comparando las placas Arduino con las ESP8266, desde el punto de vista de la comunicación, se puede apreciar que una de las características que presentan las placas tradicionales de Arduino, es su falta de conectividad al exterior. Para realizar dicha comunicación se deben utilizar módulos adicionales de comunicación de Wifi, Bluetooth, entre otros. Mientras que el microcontrolador ESP8266, trae incorporado un módulo de Wifi. Por su bajo precio lo hace ideal para proyectos de Internet de las Cosas. No obstante, no tiene una comunidad tan grande de usuarios, como los de las plataformas de Arduino. Además, presenta algunas limitaciones en cuanto a compatibilidad y uso de periféricos. Para resolver dichas falencias en 2016 Espressif, lanzó el microprocesador de doble núcleo ESP32 que amplía las capacidades de cómputo de su predecesor. Al mismo tiempo presenta un módulo WIFI y Bluetooth incorporado. Lo que permite un abanico de utilidades.

En las placas indicadas anteriormente, a excepción de la Raspberry Pi, la forma tradicional de programarlas es con el mecanismo de Bare Metal. Esto quiere decir que funcionan sin Sistema Operativo. No obstante, si el usuario lo desea, se puede instalar un Sistema Operativo, incluso de Tiempo Real, como por ejemplo FreeRTOS.

Estas placas fueron especialmente diseñadas para realizar prototipos de proyectos, pero en algunas situaciones también se los utilizan como producto final. Todas ellas siguen la premisa del proyecto Arduino, la cual consiste en que su uso sea de rápido aprendizaje, de forma tal que cualquier persona con poco conocimiento en electrónica pueda aprender a usarlas [5] [6] [7] . Muchas veces, al plantear un proyecto, un usuario común no puede saber que placa de desarrollo es adecuada para su trabajo. Para saber esto, el usuario necesita contar con el hardware físico para hacer las pruebas necesarias de funcionamiento, lo que incurre en un costo de inversión para adquirir la placa elegida. Muchas veces ocurre que el hardware seleccionado no es el adecuado para el proyecto, lo que deriva en un cambio de plataforma, conllevando así a un costo extra para la adquisición del nuevo producto, que se pueda conseguir en el mercado local. Por ese motivo, una posible alternativa para minimizar los riesgos de este impedimento, en este trabajo se investigan los emuladores para dichas placas de desarrollo.

2. LINEAS DE INVESTIGACIÓN Y DESARROLLO

Los emuladores permiten a los usuarios ejecutar en una computadora de escritorio programas de un sistema embebido, sin necesidad de contar con él físicamente. Esto se ejecuta dentro de un ambiente que imita su comportamiento [8]. A través de su utilización, una persona puede probar el sistema embebido en su computadora, antes de adquirir el hardware físico, y así verificar que le es de utilidad. Por consiguiente, teniendo en cuenta dichas características, la presente investigación se centra en construir un entorno emulado que ejecuta distintos sistemas embebidos. Todo esto funcionando con el esquema de contenedores de Docker. De manera tal, que permita realizar las pruebas de

rendimiento y las verificaciones de distintas optimizaciones de su ejecución, antes de llegar a adquirir el hardware del dispositivo.

Emulador que se utiliza en la investigación:
Existen en el mercado diferentes emuladores que imitan el comportamiento de distintos sistemas embebidos. El más utilizado es Proteus. Si bien es considerado un simulador de circuitos, también permite emular distintas placas de desarrollo, con sus actuadores y sensores de manera óptima. Este software puede emular placas de la familia Arduino, STM32, ESP8266 y Raspberry Pi. El punto negativo de este software es que su uso debe ser a través de licenciamiento pago. Por otro lado, se encuentra el simulador de Arduino Thinkercad, que es muy utilizado en ambientes educativos. La desventaja que presenta esta herramienta es que se limita únicamente a simular una placa de desarrollo Arduino UNO. Adicionalmente existe el emulador Qemu, el cual va a ser el utilizado en esta investigación. Este emulador es muy utilizado en el mercado. Entre sus principales ventajas [9], se destaca que permite trabajar con una gran cantidad de modelos de dispositivos. Además, permite detectar problemas lógicos en etapas tempranas del desarrollo, ya que brinda la depuración del dispositivo. Al mismo tiempo, presenta licenciamiento GPL. Así como también permite generar script de automatización de programas al poder ser ejecutadas desde línea de comandos. Esta característica resulta muy útil para esta investigación. De esta manera se pueden emular distintas familias de dispositivos Arduino, STM32, ESP32, ESP8266 y Raspberry Pi.

Emulador dentro de un contenedor:
Un contenedor de software es un ambiente aislado que permite compilar y ejecutar paquetes de software sin utilizar virtualización del hardware. Estos se diferencian de una máquina virtual, que realiza la completa virtualización del dispositivo físico y necesita ejecutarse sobre un Sistema Operativo que es redundante. Los contenedores se ejecutan sobre el S.O anfitrión y encima del motor de contenedores, que los emplean como base para poder funcionar. Estos contenedores a su vez pueden tener o no instalados un Sistema Operativo propio. Esta es la gran diferencia con las máquinas virtuales [10] [11].
En esta investigación se utiliza el sistema Docker, como motor de contenedores. De esta forma se está desarrollando un entorno de integración empaquetado y automatizado. Este motor va a generar contenedores de software, donde se ejecutará el emulador Qemu, que estará configurado para emular el hardware físico de las placas de desarrollo seleccionadas. El diseño de esta implementación se puede visualizar en la siguiente figura.

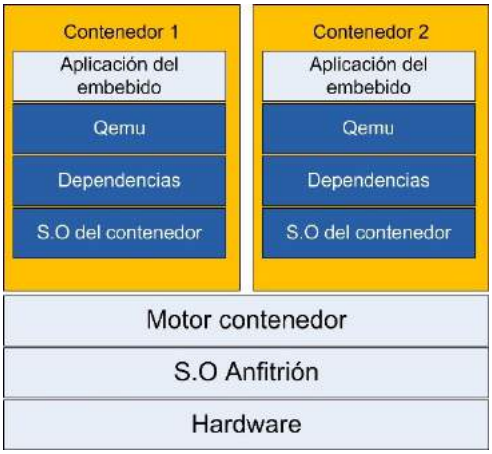

Fig. 2 Contenedor con el emulador de Sistemas embebidos

Como se puede observar en la figura, en la base de la pila se encuentra el hardware de la computadora. Sobre este funciona el Sistema Operativo Anfitrión y por encima del mismo el motor Docker. Dentro del contenedor se apilan las aplicaciones. En la parte inferior se encuentra instalado el Kernel mínimo del S.O Ubuntu. Luego se instalan las dependencias necesarias que necesita el emulador. Esto se debe a que Qemu necesita sus componentes para

poder ejecutarse. Posteriormente, en la capa siguiente se instala Qemu, ya preparado para que emule determinadas placas de desarrollo. Finalmente, en la capa superior se instala las aplicaciones del embebido, que el usuario desea ejecutar sobre la placa emulada en Qemu. De esta forma se pretende generar contenedores con imágenes de los sistemas embebidos, que permitan realizar pruebas en un entorno estandarizado, de manera tal que las aplicaciones del embebido puedan funcionar tanto en un sistema real como virtual.

3. RESULTADOS OBTENIDOS/ESPERADOS

En esta investigación se espera generar ambientes de trabajos automatizados, utilizando contenedores Docker que posean instalado y configurado el emulador Qemu, para poder utilizar distintos modelos de placas de desarrollo STM32, ESP32 y Arduino en forma virtual. De esta manera permitirá generar entornos de trabajos para automatizar pruebas de rendimiento y procesamiento simulado, de forma que los usuarios usen sus aplicaciones en el hardware físico adecuado de acuerdo con sus necesidades, incluso antes de adquirirlos.

4. FORMACIÓN DE RECURSOS HUMANOS

La presente línea de investigación dentro del departamento de Ingeniería e Investigaciones Tecnológicas forma parte del trabajo que uno de los investigadores se encuentra realizando para su maestría. Completan el grupo de investigación dos de docentes de categoría V y dos ingenieros en formación de investigador.

5. BIBLIOGRAFÍA

[1] J. Novillo-Vicuña, D. H. Rojas, B. M. Olivo, J. M. Ríos y O. C. Villavicencio, Arduino y el Internet de las cosas, Alicante: Editorial Area de Innovaciòn y Desarrollo, 2018.

[2] G. L. Nicolas Goilav, Arduino: Aprender a desarrollar para crear objetos inteligentes, Barcelona, España: Ediciones ENI, 2016.

[3] G. Trends. [En línea]. Available: https://trends.google.com/trends/explore?date=2020-01-01%202021-02-18&geo=AR&q=arduino,esp8266,raspberry%20pi,esp32,stm32.

[4] E. Carnuccio, G. De Luca, W. Valiente, M. Volker, R. Villca y M. Adagio, «Análisis de rendimiento y consumo para sistema embebido con requisitos de tiempo explícitos,» de Libro de Actas del XXVI Congreso Argentino de Ciencias de la Computación, San Justo, Universidad de La Matanza, 2020.

[5] J. Teel, How to Turn Your Arduino Project into a Sellable Product, Predictable Designs, 2016.

[6] M. Montero. [En línea]. Available: https://tecnoticias.net/2019/07/20/por-que-la-raspberry-pi-no-es-una-buena-opcion-para-productos-comerciales/#8230.

[7] D. Aranda, Electronica: Plataformas Arduino y Raspberri Pi, Buenos Aires: RedUsers, 2014.

[8] R. Milo-Sanchez, A. Fajardo-Moya y W. Rodriguez, «QEMU, una alternativa

libre para la emulación de arquitecturas de [hardware,» de V Taller de Software Libre, 2014.

[9] A. Kotovsky, How to Develop Embedded Software Using The QEMU Machine Emulator, Apriorit Inc, 2019.

[10] J. M. Ortega Candel, DOCKER. Seguridad y monitorización en contenedores e imágenes, RC-Libros, 2019.

[11] Microsoft. [En línea]. Available: https://docs.microsoft.com/es-es/learn/modules/intro-to-docker-containers/3-how-docker-images-work.