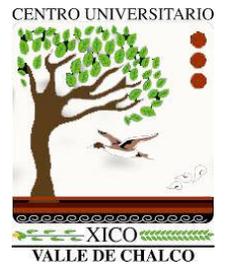




UAEM | Universidad Autónoma
del Estado de México



C.U. VALLE DE CHALCO

**DESARROLLO DE UN FRAMEWORK EN C#
PARA CONTROL DE ARDUINO UNO**

T E S I S

QUE PARA OBTENER EL GRADO DE

MAESTRO EN CIENCIAS DE LA COMPUTACIÓN

P R E S E N T A

LEONEL HERNÁNDEZ GONZÁLEZ

TUTOR ACADÉMICO
DR. SAMUEL OLMOS PEÑA

TUTORA ADJUNTA
DRA. GUOHUA SUN
TUTOR ADJUNTO
DR. JUVENAL RUEDA PAZ

VALLE DE CHALCO SOLIDARIDAD, MÉXICO SEPTIEMBRE 2018.



Valle de Chalco Solidaridad, Edo de Méx. a jueves, 14 de junio de 2018

DR. EN C. JUVENAL RUEDA PAZ
COORDINADOR DE LA MAESTRÍA CIENCIAS DE LA COMPUTACIÓN
DEL CENTRO UNIVERSITARIO UAEM VALLE DE CHALCO.

P R E S E N T E.

Por este medio le comunico a usted que la comisión revisora designada para realizar Tesis denominada: "DESARROLLO DE UN FRAMEWORK EN C# PARA CONTROL DE ARDUINO UNO", como parte de los requisitos para obtener el grado académico de Maestría en **Ciencias de la Computación** presenta **LEONEL HERNÁNDEZ GONZÁLEZ**, con número de cuenta **0922571** para sustentar el acto de evaluación de grado, ha dictaminado que dicho trabajo reúne las características de contenido para proceder a la impresión del mismo

ATENTAMENTE

Tutora adjunta

Tutor Académico

Tutor Adjunto

DRA. GUOHUA
SUN

DR. SAMUEL
OLMOS PEÑA

DR. JUVENAL
RUEDA PAZ





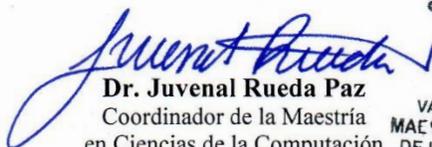
Ing. en Comp. Leonel Hernández González
Candidato al Grado de Maestría en Ciencias de la Computación
Centro Universitario UAEM Valle de Chalco
Presente

De acuerdo con el Reglamento de Estudios Avanzados de la Universidad Autónoma del Estado de México y habiendo cumplido con todas la indicaciones que la Comisión Revisora realizó con respecto a su trabajo de Tesis titulado **“Desarrollo de un framework en c# para control de arduino uno”**, la Coordinación de la Maestría en Ciencias de la Computación del Centro Universitario UAEM Valle de Chalco, concede la autorización para que proceda a la impresión de la misma.

Sin más por el momento, le reitero la seguridad de mi especial consideración y estima.

PATRIA, CIENCIA Y TRABAJO

“2018, Año del 190 Aniversario de la Universidad Autónoma del Estado de México”


Dr. Juvenal Rueda Paz
Coordinador de la Maestría
en Ciencias de la Computación
Centro Universitario UAEM Valle de Chalco


CENTRO UNIVERSITARIO
UAEM
VALLE DE CHALCO
MAESTRÍA EN CIENCIAS
DE LA COMPUTACIÓN



Valle de Chalco México a
04/09/2018

M. EN P.J. JUAN CARLOS HERNÁNDEZ HERNÁNDEZ
SUBDIRECTOR ACADÉMICO CU UAEM VALLE DE CHALCO
PRESENTE

CARTA DE AUTORIZACIÓN PARA PUBLICACIÓN EN RI

**Declaración de autoría original, inédito, libre de embargo,
susceptible de ser depositado en el Repositorio Institucional
y solicitud de evaluación profesional.**

Quien firma al calce declara que: Soy autor intelectual del original titulado "DESARROLLO DE UN FRAMEWORK EN C# PARA CONTROL DE ARDUINO UNO" y que estoy de acuerdo con la totalidad de su contenido y apruebo para ser publicado en el repositorio institucional en los términos del reglamento de acceso abierto emitido por la Universidad Autónoma del Estado de México.

Que el tipo de material presentado es original e inédito y no se encuentra en proceso de dictaminación o embargo en ninguna otra publicación o medio. Afirmando que el trabajo cumple con la cientificidad y los aspectos metodológicos para su publicación las licencias de creative commons.

Quien firma abajo solicita que el trabajo titulado "DESARROLLO DE UN FRAMEWORK EN C# PARA CONTROL DE ARDUINO UNO", sea incluido en alguna de las colecciones del Repositorio Institucional. Así mismo permitimos que la Oficina del Conocimiento Abierto realice lo propio para la conservación y difusión de la obra.

Sin otro particular

DATOS DEL AUTOR

Nombre: Leonel Hernández González

Número de cuenta : 0922571

Grado académico: Ingeniero en Computación

Programa educativo de procedencia: Maestría en Ciencias de la Computación

Institución donde labora: Independiente

Domicilio: Palmas 8, Santa María Huexoculco, Chalco, México

Teléfono/Fax: 55 17092003

Correo electrónico: leo8505@live.com.mx

A T E N T A M E N T E


Leonel Hernández González 0922571



DEDICATORIA

A mi hijo Adolfo

*Para recordarle que siempre hay algo sorprendente en la ciencia.
Sigue tus sueños, disfruta la vida, vive como tú quieras vivir, no permitas que te digan
que te equivocas si tienes la razón y si no la tienes, ten el valor para aceptarlo.*

AGRADECIMIENTOS

A mi esposa Gaby

Por impulsar mi carrera y alentarme a realizar el posgrado y porque a pesar de los tiempos difíciles, siempre estuvo a mi lado.

A mi asesor Dr. Samuel Olmos Peña.

Por su gran apoyo, sus consejos y su buen sentido del humor.

ÍNDICE DE CONTENIDO

| | |
|--|-----------|
| CAPÍTULO I. INTRODUCCIÓN | 6 |
| ANTECEDENTES | 7 |
| PLANTEAMIENTO DEL PROBLEMA | 8 |
| OBJETIVOS | 9 |
| DELIMITACIÓN DE LA INVESTIGACIÓN | 10 |
| HIPÓTESIS | 10 |
| JUSTIFICACIÓN | 10 |
| METODOLOGÍA DE LA INVESTIGACIÓN | 11 |
| CAPÍTULO II. MARCO TEÓRICO | 13 |
| IOT: LA EVOLUCIÓN DE INTERNET | 13 |
| <i>IoT en el mundo</i> | 14 |
| <i>IoT en México</i> | 14 |
| IOT: INFRAESTRUCTURA | 15 |
| <i>El Big Data</i> | 15 |
| <i>La computación en la nube y la computación en la niebla</i> | 15 |
| ARDUINO: LA SOLUCIÓN MÁS SIMPLE ES LA MEJOR | 16 |
| FRAMEWORKS: ESCRIBE MENOS, HAZ MÁS | 18 |
| ARQUITECTURA DEL SOFTWARE | 18 |
| CAPÍTULO III. METODOLOGÍA EN CASCADA | 19 |
| DEFINICIÓN | 19 |
| ANÁLISIS | 24 |
| DISEÑO | 25 |
| <i>Diagrama de secuencia</i> | 25 |
| <i>Diagrama de transición de estados</i> | 26 |
| CAPÍTULO IV. DESARROLLO DEL FRAMEWORK | 28 |
| INTRODUCCIÓN | 28 |
| CONEXIÓN DE ARDUINO CON C# USANDO PUERTO SERIE (USB) | 28 |

| | |
|--|-----------|
| ARDUINO SOFTWARE Y C#----- | 29 |
| CAPÍTULO V. RESULTADOS----- | 32 |
| USO DEL FRAMEWORK----- | 32 |
| <i>Crear proyecto</i> ----- | 33 |
| <i>Implementación del Framework</i> ----- | 34 |
| <i>Uso del Framework</i> ----- | 36 |
| <i>Método EscribirDigital(int pin, int estado)</i> ----- | 37 |
| <i>Método ConfigurarPuerto(int pin, int tipo)</i> ----- | 38 |
| <i>Método EscribirAnalogico(int pin, int estado)</i> ----- | 40 |
| <i>Método LeerAnalogico(int pin)</i> ----- | 41 |
| <i>Método Retardo(int milisegundos)</i> ----- | 43 |
| <i>Método LM35(int pin)</i> ----- | 44 |
| <i>Método HCSR04(int trig, int echo)</i> ----- | 45 |
| <i>Resumen de métodos</i> ----- | 46 |
| CONCLUSIONES Y TRABAJO FUTURO ----- | 47 |
| REFERENCIAS ----- | 49 |
| ANEXO 1 COMPARACIÓN DE CÓDIGOS----- | 52 |
| ANEXO 2 DIAGRAMAS IPO----- | 55 |
| ANEXO 3: DIAGRAMAS DE SECUENCIA ----- | 58 |
| ANEXO 4: DIAGRAMAS DE TRANSICIÓN----- | 59 |

ÍNDICE DE IMÁGENES

| | |
|--|----|
| Imagen 1.- Los beneficios de IoT. | 13 |
| Imagen 2.- Arduino y Genuino UNO. | 16 |
| 6Imagen 3.- Cargar programa a Arduino. | 32 |
| Imagen 4.- Nuevo proyecto en Visual Studio. | 33 |
| Imagen 5.- Agregar la clase Arduino UNO. | 34 |
| Imagen 6.- Administrador de dispositivos de Windows. | 35 |
| Imagen 7.- Implementación del Framework en Visual Studio. | 35 |
| Imagen 8.- Agregar botón en Visual Studio. | 36 |
| Imagen 9.- Método EscribirDigital en Visual Studio. | 37 |
| Imagen 10.- Método ConfigurarPuerto en Visual Studio. | 39 |
| Imagen 11.- Conexiones de Arduino. | 39 |
| Imagen 12.- Modulación por ancho de pulso. | 40 |
| Imagen 13.- Diseño electrónico para el uso de LeerAnalogico y EscribirAnalogico. | 42 |
| Imagen 14.- Diseño electrónico con motor y sensor de temperatura. | 42 |
| Imagen 15.- Sensor LM35. | 44 |
| Imagen 16.- Conexión del Sensor LM35 a Arduino. | 44 |
| Imagen 17.- Sensor ultrasónico HC-SR04. | 45 |

ÍNDICE DE TABLAS

| | |
|--|----|
| Tabla 1.- Interpretación de instrucciones en Arduino. | 30 |
| Tabla 2.- Equivalencia de pines analógicos en el método EscribirDigital. | 37 |

ÍNDICE DE CÓDIGOS

| | |
|--|----|
| Código 1.- Estructura básica de un programa en Arduino Software. | 29 |
| Código 2.- Comunicación serial desde Arduino. | 29 |
| Código 3.- Intercambio de datos desde Arduino. | 30 |
| Código 4.- Comunicación serial desde C#. | 31 |
| Código 5.- Implementación de la clase ArduinoUNO en C#. | 34 |
| Código 7.- Método EscribirDigital(). | 36 |
| Código 8.- Equivalencia de pin analógico en el método EscribirDigital(). | 38 |
| Código 9.- Método ConfigurarPuerto(). | 38 |
| Código 10.- Método LeerAnalogico(). | 41 |
| Código 11.- Asignar valor de sensor a led. | 43 |
| Código 12.- Asignar valor de sensor a motor. | 43 |
| Código 13.- Asignar valor equivalente. | 43 |
| Código 14.- Uso del método Retardo(). | 43 |
| Código 15.- Uso del método HCSR04 de C#. | 45 |

ÍNDICE DE FIGURAS

| | |
|--|----|
| Figura 1.- Actividades del ciclo de vida clásico o en cascada. | 19 |
| Figura 2.- Modelo lineal o en cascada. | 20 |
| Figura 3.- Modelo en cascada. | 20 |
| Figura 4.- Etapas del modelo en cascada. | 21 |
| Figura 5.- Secuencia de actividades para el modelo de cascada. | 22 |
| Figura 6.- Modelo en cascada. | 23 |
| Figura 7. Diagrama de secuencia para el manejo de puertos. | 26 |
| Figura 8. Diagrama de transición de estados para el manejo de puertos. | 27 |
| Figura 9.- Comunicación entre C# y Arduino. | 28 |

RESUMEN

El internet de las cosas se define de forma general, como la interacción de objetos de la vida cotidiana con los seres humanos y es uno de los temas de investigación más importantes de nuestros tiempos, sin embargo, en México no se tiene el avance que se espera de un país con su nivel de desarrollo. Las razones de esto son la baja divulgación de este tema en las universidades y en general en las escuelas; el nivel de complejidad, ya que se requiere conocer temas de programación, electrónica, sistemas digitales y comunicaciones y el tiempo de desarrollo, debido a que es necesario escribir código en dos lenguajes diferentes, el de la placa de desarrollo y otro de alto nivel.

Este trabajo se centra en mostrar a los profesionales de la computación y a los interesados en el uso de IoT la forma de desarrollar aplicaciones con el uso de la placa Arduino y C#, además, como resultado del trabajo de investigación, se genera un marco de trabajo para la elaboración rápida de aplicaciones de este tipo.

ABSTRACT

The Internet of things is the interaction among of objects of daily life with human and it is one of the most important research topics of our time. However, in Mexico there is not the expected progress of a country with its level of development. The three main reasons for this are; i) the low disclosure of this topic in the universities and in general in the schools, ii) the level of complexity, It is necessary to know topics about of programming, electronics, digital systems and communications and iii) the development time.

This research work focuses on showing computer area professionals and those interested in the use of IoT the way to develop applications with the use of Arduino and C #, in addition, because of the research work, a framework is generated for the rapid development of applications of this type.

Capítulo I. INTRODUCCIÓN

El presente trabajo de investigación se divide en cinco capítulos, el tratamiento de cada uno de estos se soporta y complementa en figuras, cuadros, tablas y gráficas, con la finalidad de facilitar la lectura y comprensión, el contenido de estos se describe brevemente a continuación:

Capítulo I, Introducción. Es un breve preámbulo al trabajo y la metodología de investigación que lo sustenta, así mismo, enfatiza la importancia del uso de frameworks y el manejo de las placas como Arduino.

Capítulo II, Marco teórico. Aborda la definición e historia del internet de las cosas y su estado actual tanto en México como en el mundo, se describe la placa de desarrollo y se justifica el uso del modelo Arduino Uno, finalmente se presentan algunos ejemplos de los frameworks existentes.

Capítulo III, Metodología del modelo en cascada. Se plantea el modelo de desarrollo de software utilizado en la construcción del framework. Describe detalladamente cada uno de los métodos y procedimientos que se realizan en este trabajo.

Capítulo IV, Desarrollo de los módulos del framework. Se presenta el desarrollo en lenguaje de programación C# de los diferentes módulos para la placa. Todos estos se desarrollan basados en la metodología de desarrollo en cascada.

Capítulo V, Resultados Experimentales. Se presentan los métodos resultantes del framework propuesto, así como su funcionalidad y estructura.

Conclusiones y trabajo futuro. Determinan el cumplimiento de los objetivos planteados inicialmente, también se incluyen las recomendaciones y trabajo futuro, tales como, el desarrollo de un framework para controlar la placa Arduino desde Android a partir del firmware incluido en este framework.

Finalmente se incluye la bibliografía y 4 anexos, el primero muestra una comparación del número de líneas necesarias para realizar una sencilla aplicación con el framework desarrollado en este trabajo y sin él, los otros 3 anexos presentan diagramas relacionados a la construcción del framework.

ANTECEDENTES

Arduino es una placa de desarrollo programable que cuenta con su propio lenguaje de desarrollo (Arduino Software) similar al lenguaje de programación C, con el cual, se elaboran rutinas que serán ejecutadas directamente por su microcontrolador ATmega328, cuenta con 14 pines digitales de entrada/salida, 6 entradas analógicas, un oscilador de cristal de 16MHz, una conexión USB, un conector de alimentación, entre otras cosas (Reyna et al, 2014).

Esta placa de desarrollo puede utilizar lenguajes de alto nivel como C#, Java y Python entre otros, para conectarse con computadores que recogen datos desde sensores conectados a esta placa de desarrollo. Los datos pueden ser procesados directamente por el dispositivo o por la computadora a la cual está conectado.

La gama de estas tarjetas desarrollo se compone de Arduino Zero, Uno, Leonardo, 101, Esplora, Mega, Due y M0 Pro. En el desarrollo del framework se utiliza la serie Uno, la cual, puede ser utilizada como objeto de estudio por su relativa sencillez con respecto a otras series de gama alta, bajo costo, abundante documentación y ejemplos sencillos (Arduino, 2015).

Por su parte C# es el lenguaje utilizado para el desarrollo de aplicaciones de Windows y éste a su vez, es el sistema operativo para computadoras de mayor uso en el mundo con el 90.54% según NetMarket (2015). C# es un lenguaje similar a Java, una evolución de Microsoft C++ con la finalidad de adaptarse al paradigma orientado a objetos, robusto y seguro, usado tanto para aplicaciones de escritorio como para web (Price & Gunderloy,2002).

Los proyectos con estas tarjetas y Windows se realizan programando en el software de Arduino una rutina que recibirá parámetros desde el lenguaje de alto nivel, en este punto

cada desarrollador implementa los métodos que considere necesarios, después, generalmente realiza una clase en C# que servirá como puente para la comunicación con la placa, aunque, es posible omitir esta clase e implementarla directamente en la capa superior (Kurniawan, 2012).

El procedimiento descrito anteriormente consume una gran cantidad de tiempo, se pueden presentar errores que terminan aplazando los proyectos, el código final es ilegible para otros desarrolladores y en ocasiones incluso para la persona que lo creó, por esto, es necesario crear un procedimiento que realice la comunicación entre la capa superior de la aplicación y la placa.

El framework que se desarrolla busca reducir los tiempos de programación de aplicaciones que involucren el uso de Arduino y C#, de tal manera, que el desarrollador pueda enfocarse a crear la funcionalidad que requiere y con el ahorro de tiempo poder ampliar o mejorar sus aplicaciones.

Con esta mejora se pretende motivar a los estudiantes y profesionales en computación, electrónica, robótica, mecatrónica, entre otros, a la creación de tecnología emergente. El uso del framework en combinación con las API's de Windows dan como resultado Arduino conectado a internet, controlado por mandos de Xbox o por voz con la API de Cortana.

PLANTEAMIENTO DEL PROBLEMA

La elaboración de aplicaciones para Arduino con comunicación con otros dispositivos ya sea, teléfonos inteligentes o computadoras por lo general consume mucho tiempo de desarrollo, debido a que es necesario implementar métodos para conectarse con la placa y para manipular datos.

La falta de métodos que faciliten la elaboración de programas para Arduino limita el desarrollo de aplicaciones, debido a que se requiere de mucho tiempo para aprender el lenguaje de la placa de desarrollo y es necesario realizar clases especiales para cada proyecto. Esto obstaculiza que los estudiantes y profesionales de computación, robótica y mecatrónica realicen desarrollos avanzados para este hardware, quedándose en implementaciones básicas como verificar el estado de un puerto, encender y apagar un led.

A lo anterior se puede agregar que el código resultante puede llegar a ser ilegible y por lo tanto no reutilizable, lo que se traduce en más tiempo de desarrollo en cada proyecto nuevo.

En el Anexo 1 se muestra la diferencia entre el número de líneas de código que se requieren para crear un programa en C# para el control de una placa de manera tradicional y usando el framework desarrollado en este trabajo de tesis. El resultado es de 31 líneas con el uso tradicional y 4 con el framework, lo cual representa una reducción del 87.1%.

La pregunta de investigación que surge para el presente estudio es la siguiente:

¿Cuáles son los métodos que debe contener un framework eficiente que logre minimizar el tiempo de desarrollo de aplicaciones que involucren C# y la placa Arduino Uno?

OBJETIVOS

General

Crear un framework en C# y Arduino Software que simplifique y agilice el proceso de programación de aplicaciones que involucren un dispositivo con Windows y una placa Arduino.

Específicos

- i. Realizar el análisis para el desarrollo del software.
- ii. Diseñar los diagramas para la construcción de software.
- iii. Crear las clases necesarias en Arduino Software para comunicarse con la placa de desarrollo.
- iv. Desarrollar las clases necesarias para controlar Arduino desde C#.
- v. Verificar el funcionamiento de las clases implementadas.
- vi. Elaborar un manual técnico y del usuario.

DELIMITACIÓN DE LA INVESTIGACIÓN

La placa utilizada es Arduino Uno R3, es decir, la revisión número tres, porque es la versión más simple de Arduino, más barata y es representativa de todas las otras placas Arduino, debido a que las demás series tienen características similares. La revisión 3 corresponde a la más actual a la fecha. Esta placa es de fácil acceso para estudiantes y aficionados gracias a su bajo costo.

El lenguaje utilizado es C# versión 7, ya que es el lenguaje de programación para el desarrollo de aplicaciones para Windows, éste a su vez, es el sistema operativo para computadoras más usado en el mundo.

El compilador utilizado fue Visual Studio 2017 Community, debido a que es de uso libre y se utiliza para compilar aplicaciones de C#, cuenta con herramientas de depuración y pruebas de rendimiento de software.

La implementación y pruebas se realizan en una PC con Windows 10 Pro, esta es la versión reciente del sistema operativo lanzado el 31 de Julio de 2015, es posible programar aplicaciones universales que funcionen tanto en PC, teléfonos inteligentes e incluso la consola de videojuegos Xbox One.

Se crean métodos en C# para el manejo de los sensores LM35 y HCSR04 que tienen una configuración y una conexión a puertos configurables, esto con la finalidad de mostrar la forma de ampliar el framework con otros sensores.

HIPÓTESIS

Se puede construir un framework en C# que sea capaz de abstraer los métodos necesarios para el control de una placa Arduino. Los métodos resultantes reducirán el número de líneas de código necesarias para la elaboración de aplicaciones de este tipo.

JUSTIFICACIÓN

El presente trabajo está elaborado para simplificar la programación de aplicaciones en C# para el control de Arduino Uno R3, ya que en la actualidad este proceso, como se mencionó anteriormente, es complicado y requiere de mucho tiempo, por lo tanto, las aplicaciones se vuelven limitadas.

Es necesario agilizar el proceso de creación de software para el control de placas de desarrollo, con la finalidad de generar aplicaciones más completas en domótica, robótica y mecatrónica. El desarrollo del framework motivará a estudiantes y profesionales de las disciplinas mencionadas y de otras a abordar nuevos proyectos con Arduino. Este prototipo de framework permitirá desarrollar más y mejores aplicaciones para Arduino en C# en menos tiempo.

METODOLOGÍA DE LA INVESTIGACIÓN

Dependiendo del problema de investigación, la metodología para abordar este, hace uso de diferentes aproximaciones este trabajo de tesis para su desarrollo, se sustenta en tres etapas que se sustentan en diferentes tipos (Tamayo, 2004) y se mencionan a continuación:

1. Etapa documental, basada en consultas bibliográficas y hemerográficas sobre los trabajos realizados hasta el momento.
 - a. Se definen los conceptos de IoT de acuerdo con diferentes fuentes. Se forma una sola definición a partir de los conceptos analizados.
 - b. Se investiga la situación actual del IoT en el mundo y en México, así como, los beneficios de su utilización.
 - c. La búsqueda de dispositivos para IoT muestran a Arduino como el más sencillo de utilizarse, así que, se realiza una investigación acerca de su estructura y funcionamiento.
 - d. La investigación de Arduino conlleva al uso de Arduino IDE, por lo tanto, se hace un estudio detallado de las principales funciones y características.
 - e. Se establece que para ser considerado IoT, el dispositivo debe tener una comunicación con otro, usando cualquier protocolo.
 - f. Lo anterior requiere de otro dispositivo, por lo tanto, se elige una computadora.
 - g. La investigación muestra que la mayor parte de las computadoras usan Windows.
 - h. C# es el lenguaje con mayor impacto en el desarrollo de aplicaciones para Windows, de acuerdo con los datos obtenidos en esta investigación.

- i. Se realiza un estudio del lenguaje C# y su comunicación con otros dispositivos.
 - j. El resultado de la investigación documental muestra compatibilidad entre el lenguaje de programación y el dispositivo.
 - k. Se estudian los tipos de metodologías y se elige la que más se adapta al tipo de proyecto.
2. Etapa de aplicación, al utilizar estudios y trabajos realizados anteriormente con la finalidad de desarrollar algo nuevo. Se desarrolla un software utilizando la metodología en cascada adaptada. La aplicación se presenta en los capítulos III y IV.
3. Etapa de síntesis. El objetivo de esta etapa es analizar y mostrar los resultados obtenidos. los resultados de esta etapa se presentan en el Capítulo V. Finalmente, la discusión, conclusiones, implicaciones del estudio y trabajo futuro componen la última sección del trabajo

Capítulo II. MARCO TEÓRICO

IoT: LA EVOLUCIÓN DE INTERNET

El internet de las cosas (IoT, por sus siglas en inglés) es una nueva forma de comunicación entre objetos de la vida cotidiana con los seres humanos. Conocido en un principio como Cómputo ubicuo o computación ubicua, es ahora el centro de los temas de investigación y de acuerdo con Gartner Consulting (2017) para 2020 más del 65% de las empresas habrán adoptado el IoT a sus actividades. Mejoras en el bienestar y en los servicios, generación de ingresos y mayor seguridad son algunos de los beneficios de IoT de acuerdo con la Imagen 1 tomada y traducida del libro electrónico Leading the IoT de Gartner Consulting.

EL INTERNET DE LAS COSAS



Imagen 1.- Los beneficios de IoT. (Adaptación de Gartner Consulting, 2017).

Una definición formal de IoT es que se trata de red de objetos físicos (cosas) que contienen incrustados tecnología para comunicarse y sensor o interactuar con sus estados internos o el ambiente externo. La conexión de objetos, procesos y personal habilita la captura de datos y eventos de los cuales una compañía puede aprender el comportamiento y el uso, reaccionar con una acción preventiva, o aumentar o transformar procesos comerciales. El IoT es una capacidad fundamental para la creación de un negocio digital (Gartner Consulting, 2017).

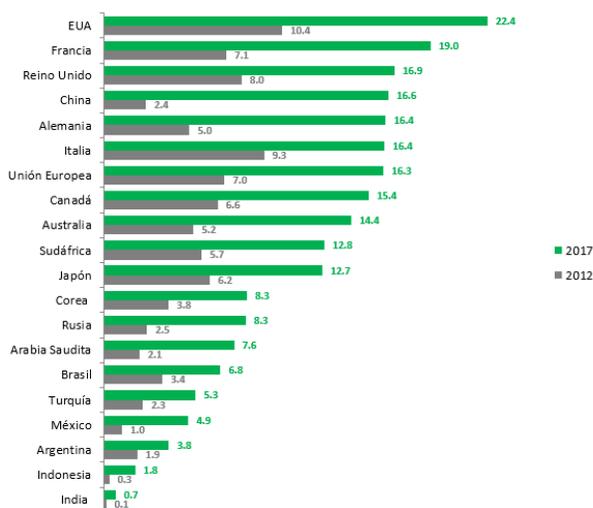
IoT en el mundo

El IoT debería ser tan fácil como activar un interruptor, expresó Hyunsuk Kim jefe de la división de electrónica de consumo de Samsung y Samsung Research en el CES 2018 al presentar la app SmartThings, una aplicación que unirá Samsung Connect, Smart Home, Smart View y otras tecnologías Samsung para ser controlados desde un teléfono inteligente, en este evento en el cual las compañías presentan sus avances tecnológicos Hyunsuk Kim afirmó que es necesario crear estándares en IoT para continuar el desarrollo de esta tecnología (Forbes, 2018).

En el mismo evento Tim Baxter presidente de Samsung Norte América señaló que el 90% de los electrónicos y televisores de esta compañía están conectados al Internet de las Cosas y que para el 2020 estaremos viviendo en una realidad totalmente conectada.

IoT en México

Mientras tanto en México en avance en IoT se encuentra rezagado, la gráfica 1 tomada de The Social Intelligence Unit (2017) muestra la posición de este país con relación a los otros integrantes de la Organización para la Cooperación y el Desarrollo económicos (OCDE) del Grupo de los 20 al cual pertenece.



Gráfica 1.- Conexiones M2M por cada 100 habitantes. (The Social Intelligence Unit, 2017)

En la gráfica anterior se presenta las conexiones machine-to-machine (M2M) que sirve como referencia para el IoT de cada país. México se muestra por debajo del promedio y casi al final de la lista del Grupo de los 20.

La construcción de un framework y en general de herramientas que faciliten el desarrollo de aplicaciones de IoT motiva a estudiantes, profesionales y aficionados mexicanos a experimentar y crear más aplicaciones.

IoT: INFRAESTRUCTURA

El nacimiento del internet de las cosas trae consigo el desarrollo de diferentes componentes necesarios para su ejecución, entre los cuales están:

El Big Data

De acuerdo con Cisco (2015), se estima que para 2020 se encuentren conectadas 50 billones de cosas a internet generando datos y enviándolos a la nube para ser procesados. Los grandes bancos de datos, es decir el Big Data será el encargado de almacenar y procesar esta información.

La computación en la nube y la computación en la niebla

La computación en la nube es la manera de referirse al conjunto de servidores conectados a internet recibiendo datos de clientes para ser procesados y entregar un resultado, evitando así, el procesamiento y almacenamiento local.

Según Cisco, en 2016 el tráfico diario de datos fue de 2 exabytes, por lo cual, considera que el modelo actual de computación en la nube no será suficiente para el Internet de las Cosas y propone un nuevo modelo llamado Computación en la niebla (Fog Computing), el cual consiste en que las cosas conectadas a internet, tales como, refrigeradores, semáforos, autos, etc., procesen los datos recibidos por sus sensores y únicamente envíen a los servidores los resultados cuando sea necesario su almacenamiento o requieran de terceros.

ARDUINO: LA SOLUCIÓN MÁS SIMPLE ES LA MEJOR

El principio de parsimonia (también conocido como navaja de Occam u Ockham) establece que teniendo varias soluciones la mejor siempre será la más simple. En términos de programación se traduce como “menos programación es mejor” y es en esto donde Arduino es líder y la placa Arduino UNO es la más simple.

Ciertamente Arduino no es una placa de alto desempeño, que se utilice en las industrias o que sea la mejor, sin embargo, es barata, accesible a cualquier estudiante de nivel superior o medio superior, cuenta con una vasta documentación, ejemplos y accesorios. Lo anterior la hace ideal para practicar y aprender acerca del internet de las cosas. La imagen 2 fue tomada de la página oficial de Arduino y muestra una placa Arduino Uno y su equivalente en Europa Genuino Uno.



Imagen 2.- Arduino y Genuino UNO. (Arduino,2018).

La definición formal de Arduino UNO de acuerdo con la página oficial es que se trata de una plataforma computacional física open-source basada en una simple tarjeta de I/O y un

entorno de desarrollo que implementa el lenguaje Processing/Wiring (Arduino,2018). Arduino UNO cuenta con las siguientes características:

- Microcontrolador ATmega328.- Pertenece a la familia de microcontroladores AVR de 8 bits de alto desempeño (Microchip,2018) y tiene las siguientes propiedades.
 - a. Flash: 32 Kbytes
 - b. SRAM: 2 Kbytes
 - c. Cantidad Pines: 28
 - d. Frecuencia máxima de operación: 20 MHz
 - e. CPU: 8-bit AVR
 - f. Pines máximos de E/S: 23
- Voltaje de entrada 7-12V.
- 14 pines digitales de I/O (6 salidas PWM). Enumerados del 0-13 con 5v y 40mA, además algunos pueden adoptar un propósito específico, tales como:
 - a. Rx y Tx para recibir y transmitir datos.
 - b. 2 y 3 para interrupciones.
 - c. PWM (~) 3,5,6,9,10 y 11 con señales modulares por ancho de pulso.
- 6 entradas analógicas. Enumeradas de A0 a A5 con 10 bits de resolución de 0 a 5v.
- 32k de memoria Flash.
- Reloj de 16MHz de velocidad.

El Framework que se construye en este trabajo contiene métodos en C# equivalentes a las funciones existentes en Arduino Software, de esta manera no será necesario editar el código de Arduino, sino que bastará con usar las de C# para obtener una funcionalidad similar. Las funciones que tienen equivalencia son: `digitalWrite()`, `digitalRead()`, `pinMode()`, `analogRead()`, `analogWrite()` y `delay()`. Estos métodos se encuentran en la clase de C# con nombres en español para facilitar su aprendizaje a desarrolladores de habla hispana.

Además de Arduino, existe otras placas de desarrollo, tales como: Raspberry Pi 3, BeagleBone Black, Joule 570X de Intel entre otras.

FRAMEWORKS: ESCRIBE MENOS, HAZ MÁS

JQuery es uno de los Frameworks más conocido en el mundo, ha sido creado para ahorrar tiempo al escribir código de JavaScript, su eslogan “Escribe menos, haz más” describe la finalidad de cualquier Framework.

El presente trabajo no pretende crear un Framework del tamaño de JQuery, sino presentar las bases para crear algo similar, que se aplique al desarrollo de programas de IoT con Arduino, escritas en C#.

En la actualidad se pueden encontrar diferentes Frameworks para cada lenguaje: Laravel, Zend y Symfony para PHP; Angular, JQuery y Modernizr para JavaScript; Spring MVC, JSF y Vaadin para Java; Django, Flask y Pyramid para Python, entre muchos otros.

Galindo y Camps (2008) definen un Framework como “un conjunto de clases cooperativas que construyen un diseño reutilizable para un tipo específico de software. Un Framework proporciona la arquitectura partiendo el diseño en clases abstractas y definiendo sus responsabilidades y colaboraciones. Un desarrollador realiza una aplicación haciendo subclases y componiendo instancias a partir de las clases definidas por el Framework”.

La definición anterior debe ser modificada actualmente para incluir Frameworks que están contruidos en una sola clase o que se encuentran en un solo archivo, tales como, JQuery o Bootstrap.

La característica principal de un Framework es la inversión de control, ya que mientras en una biblioteca de clases el control de flujo se encuentra en la aplicación, en el Framework es el que realiza las llamadas y contiene la funcionalidad, esto hace que el desarrollador no tenga la necesidad de saber cómo está construido.

ARQUITECTURA DEL SOFTWARE

Se ha optado por elaborar este desarrollo con la metodología tradicional en cascada, ya que se requiere de diagramas para su futura ampliación o modificación, los cuales no se elaboran en otras metodologías.

Los diagramas de la arquitectura del software se encuentran en el Anexo 1 de ese trabajo.

Capítulo III. METODOLOGÍA EN CASCADA

Desde la década de 1970 se han utilizado las metodologías estructuradas para documentar, analizar y diseñar sistemas de información. La metodología en cascada es de forma estructurada, en la cual, cada actividad se construye con base a la anterior, por lo tanto, esta metodología se sigue de arriba hacia abajo, avanzando desde el nivel más alto con menos detalle, hasta lo más específico.

Esta metodología es lineal, es decir, para poder iniciar la siguiente fase, la anterior debe estar terminada, por lo tanto, este ciclo puede consumir mucho tiempo, dependiendo de la experiencia de los desarrolladores o desarrollador para definir entradas, procesos y salidas del sistema (Laudon y Laudon, 2004).

DEFINICIÓN

La metodología en cascada pertenece a las metodologías tradicionales y a lo largo de la historia diferentes autores han elaborado las definiciones que a continuación se presentan:

“El modelo en cascada suele ser utilizado cuando el sistema no es de gran complejidad y puede ser manejable como proyecto, cuando los requerimientos del sistema pueden predecirse fácilmente” (Taboada y Cotos, 2005).

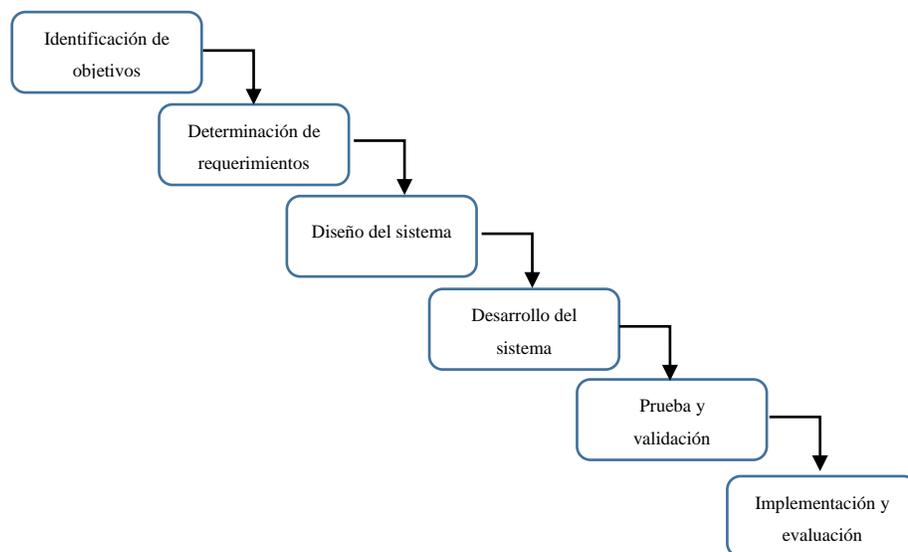


Figura 1. Actividades del ciclo de vida clásico o en cascada (Taboada y Cotos, 2005).

Para estos autores este modelo consiste en una sucesión de etapas en cadena o cascada por las que pasa un sistema de información desde que es concebido hasta que comienza a usarse. Las etapas que proponen se pueden visualizar en la Figura 1.

“Es donde no se inicia una etapa o fase hasta que se completa la anterior, cada vez que finaliza una etapa se obtiene un documento o producto final, que, revisado, validado y aprobado, sirve como aproximación y documentación de partida para la siguiente” (Barranco, 2003). Esta metodología es la más extendida y utilizada en proyectos de gestión, medianos y grandes. Este autor propone solo cuatro etapas (ver figura 2).

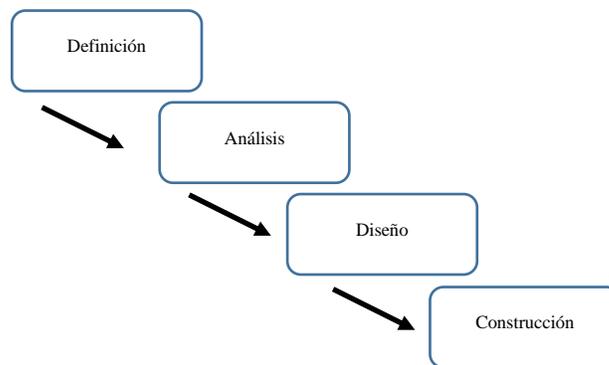


Figura 2. Modelo lineal o en cascada (Barranco, 2003).

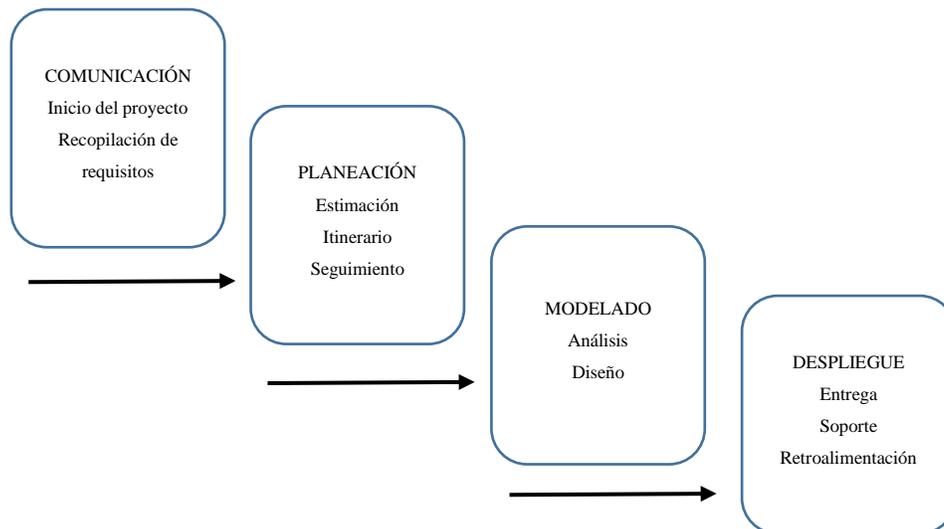


Figura 3. Modelo en cascada (Pressman, 2005).

Pressman (2005) explica que el modelo en cascada sugiere un enfoque sistemático, secuencial hacia el desarrollo del software, que según para este autor se inicia con la especificación de requerimientos del cliente y que continúa con la planeación, el modelado, la construcción y el despliegue para culminar en el soporte del software terminado (ver figura 3).

La representación de este modelo se muestra en la figura 4, el cual representa la cascada de una fase a otra, en principio, “el resultado de cada fase es uno o más documentos aprobados, la siguiente fase no debe empezar hasta que la fase previa haya finalizado, estas etapas se superponen y proporcionan información a las otras” (Sommerville, 2005).

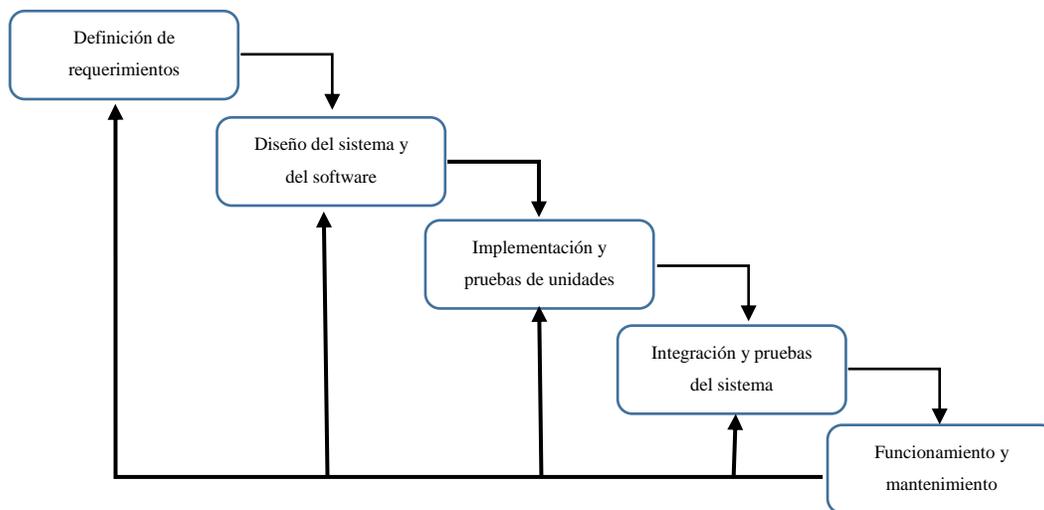


Figura 4. Etapas del modelo en cascada (Sommerville, 2005).

Para este autor el proceso de esta metodología no es un modelo lineal simple, sino una serie de iteraciones de las actividades de desarrollo. Estas iteraciones son costosas y además implican rehacer el trabajo, lo que lo hace un modelo lento, sin embargo, el software resultante contiene menos errores y se encuentra mejor documentado.

El modelo en cascada, también conocido como secuencial, es una metodología para la elaboración de software que se desarrolló entre las décadas de los sesenta y setenta (Weitzenfeld, 2005). Este modelo es el más arcaico para la elaboración de software que consiste en una secuencia de pasos bien definidos (algo que rara vez se tiene en la elaboración de software), uno tras otro, haciendo entregas en cada uno conocidas como hitos o checkpoints (ver figura 5).

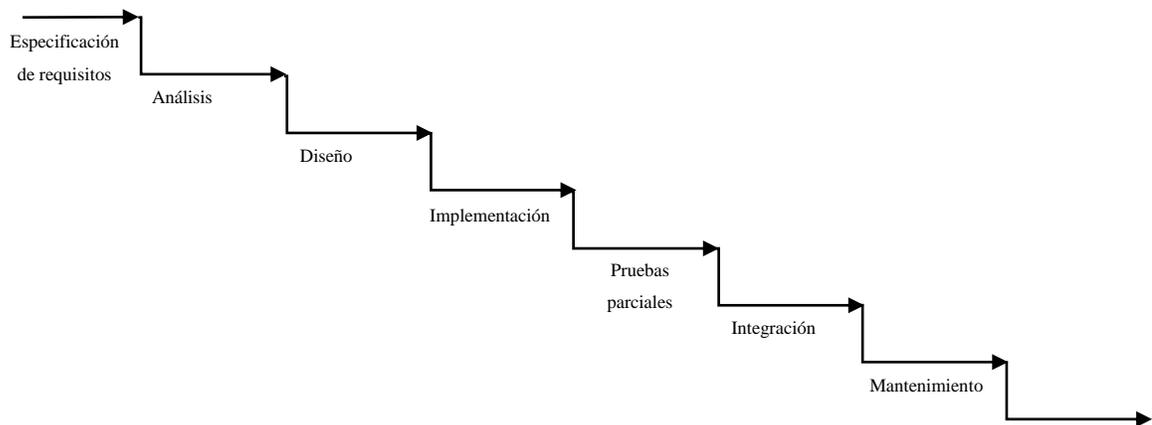


Figura 5. Secuencia de actividades para el modelo de cascada (Weitzenfeld, 2005).

La figura 5 describe el orden de las actividades del desarrollo de software, la cual no muestra una etapa explícita de documentación porque se lleva a cabo en el transcurso de todo el desarrollo, el autor ha realizado una modificación al modelo, debido a que, el original plantea que en cada una de sus actividades debería completarse antes de poder continuar con la siguiente, sin embargo, ha extendido el modelo permitiéndole regresar a las etapas anteriores.

Ganzábal plantea un modelo en cascada compuesto por 6 etapas: análisis, diseño, implementación, pruebas, implantación y mantenimiento. Estas etapas se muestran en la figura 6 y son similares a los descritos por los otros autores, la diferencia se encuentra en que Ganzábal ha simplificado los nombres de cada etapa.

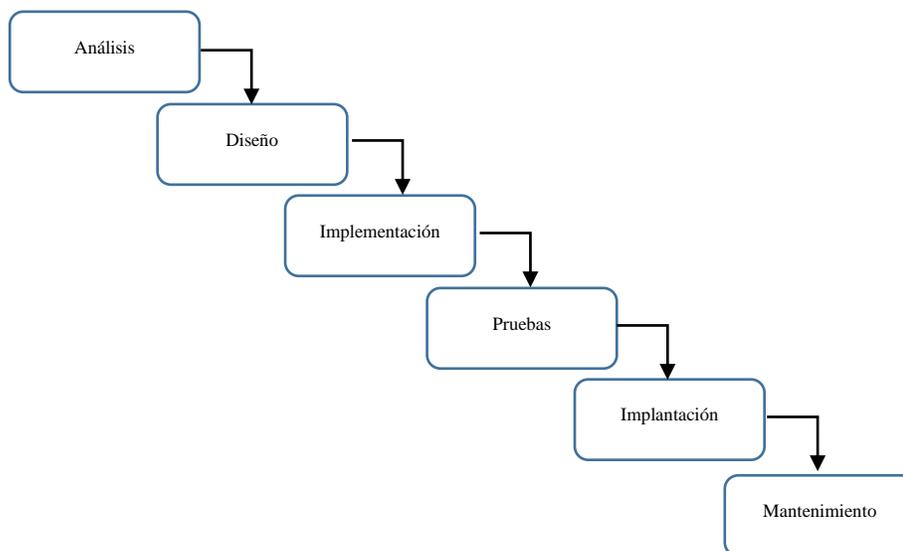


Figura 6. Modelo en cascada (Ganzábal, 2014).

En este punto, es preciso hacer una pausa para justificar el uso de la metodología en cascada en el desarrollo de esta tesis, ya que, seguramente se cuestionará la utilización de una metodología tradicional de los años 70's en un trabajo para elaborar tecnología nueva y más aún, una metodología estructurada para un lenguaje que nació siendo orientado a objetos.

Un trabajo que requiera de documentación para ser modificado o ampliado debe realizarse con metodologías que brinden una estructura rígida en la elaboración de software. Cualquier aplicación que sea de código abierto, como lo es el caso de este trabajo, debe contener una documentación que permita a otros programadores continuarlo, corregirlo o ampliarlo, por lo tanto, una metodología tradicional es apropiada para la elaboración de un framework de código abierto.

La metodología en cascada se puede utilizar en los lenguajes orientados a objetos si en la fase de diseño se realiza una adaptación de los diagramas utilizados en esta metodología a sus equivalentes de las metodologías ágiles, como se muestra en este trabajo con los diagramas de secuencia en el Anexo 3.

La principal ventaja de las metodologías ágiles es la velocidad, proporcionada por la división de tareas (Springs) entre los desarrolladores, permitiendo así un trabajo en paralelo. En una aplicación realizada por una sola persona esta ventaja es nula, por lo tanto, la mejor opción es la metodología en cascada.

ANÁLISIS

El primer paso para la elaboración del software es el análisis de la aplicación que según Kendall (2005), es la identificación de las necesidades del usuario y la determinación de los requisitos que deberá contener la aplicación. Por lo tanto, es la unión de los requisitos especialmente del software, para que el desarrollador del software interprete de manera fácil la naturalidad de los programas que deberán realizar para el avance de este, entre el desarrollo de este paso se deberá analizar el comportamiento, rendimiento, funciones requeridas y la interconexión.

Para poder terminar con éxito un proyecto de software, es de suma importancia cuidar su desarrollo, porque antes de que se empiece a codificar se debe tener una completa comprensión de sus requisitos. Pressman fundamenta que el trabajo del análisis de requisitos es descubrir, modelar y especificar los detalles del proyecto, en esta parte se crean modelos de los requisitos de datos, flujo de información y control, gracias a esto se pueden analizar soluciones para asignar diferentes elementos del software, esto permite que un desarrollador o desarrolladores especifiquen funciones para un mayor rendimiento de este.

A continuación, se muestran los requerimientos para la elaboración del framework a partir del análisis de los requisitos que debe de cumplir una vez desarrollado.

1. Debe haber conexión entre ambos una computadora y la placa.
2. Los eventos deben poder ser implementados desde C#.
3. El programador no debe tener la necesidad de modificar el código de Arduino a menos que requiera ampliar las funciones.
4. Los métodos resultantes deben ser fácil de usar y de aprender.
5. El tiempo de implementación de las clases debe ser el mínimo posible.

DISEÑO

El diseño del software es una representación conceptual de las necesidades del usuario y la arquitectura deseada del software (Weitzenfeld, 2005). Una vez que se ha completado el análisis de requerimientos, se procede a la elaboración de los diagramas que mostrarán la arquitectura del software, existen diferentes metodologías y herramientas para abordar esta etapa.

Para la etapa de diseño del modelo en cascada se utilizan diagramas de tipo estructurado como son Diagrama de secuencia, Diagrama de flujo de datos, Diagrama de modelo de datos, Diagrama de transición de estados, entre otros. Es importante mencionar que estos diagramas se utilizaron desde la creación del modelo en cascada para los lenguajes de programación estructurada, sin embargo, con la llegada del paradigma orientado a objetos, han cambiado también los diagramas utilizados, en la actualidad, se pueden usar los Diagramas de clase y otros del nuevo paradigma.

En el presente trabajo se describen los diagramas propios del modelo en cascada, la implementación se realiza con la herramienta CASE Microsoft Visio 2013 Profesional.

Diagrama de secuencia

El diagrama de secuencia se define como un esquema conceptual que permite representar el comportamiento de un sistema, es decir, muestra la forma en la cual se realizan los procesos utilizando objetos dentro de un escenario y los mensajes intercambiados entre ellos (Zapata y Garcés, 2008). A continuación, se muestra el diagrama de secuencia para el manejo de puertos, el resto de este tipo de diagramas se encuentran en el Anexo 3.

DIAGRAMA DE SECUENCIA PARA EL MANEJO DE PUERTOS

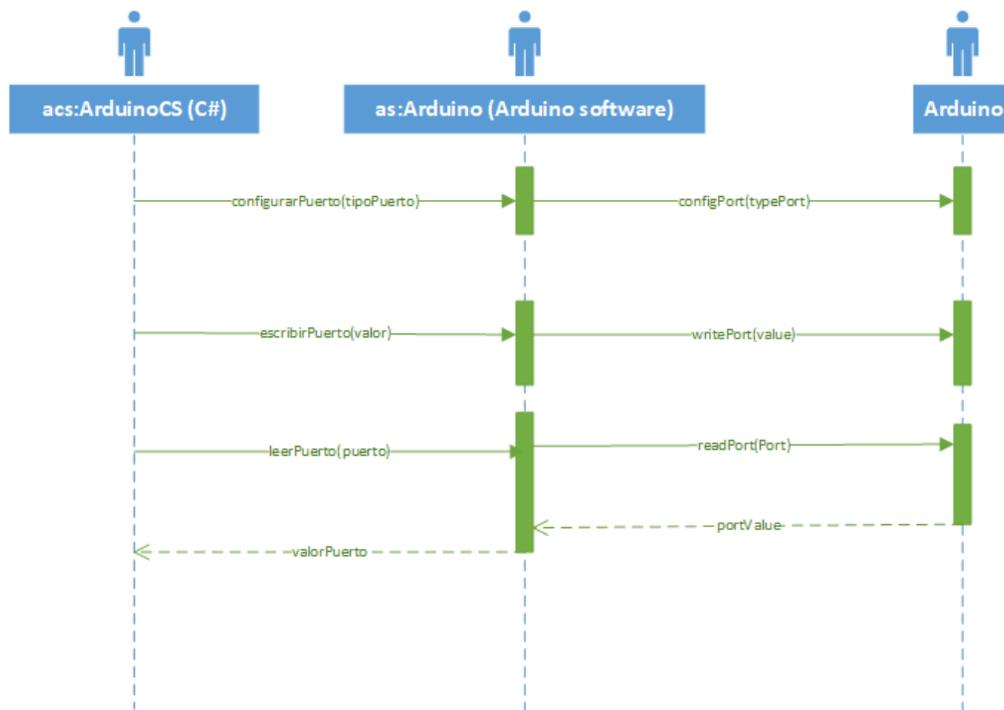


Figura 7. Diagrama de secuencia para el manejo de puertos (Elaboración propia, 2016).

Diagrama de transición de estados

Los diagramas de estado describen el comportamiento de una aplicación a través de los procesos que se realizan, a los cuales se les conoce como estados del sistema, cada estado está dirigido hacia otro o a un estado final (Fowler, 2004). El diagrama de transición de estados consta de un estado inicial que indica el comienzo de los procesos, varios estados que representan un instante de vida de un objeto, estados de transición que pasa de un estado a otro y un estado final. En la figura 8 se muestra un diagrama de transición de estados para el manejo de puertos. En el Anexo 4 se proporcionan otros diagramas de este tipo.

DIAGRAMA DE TRANSICIÓN DE ESTADOS PARA EL MANEJO DE PUERTOS

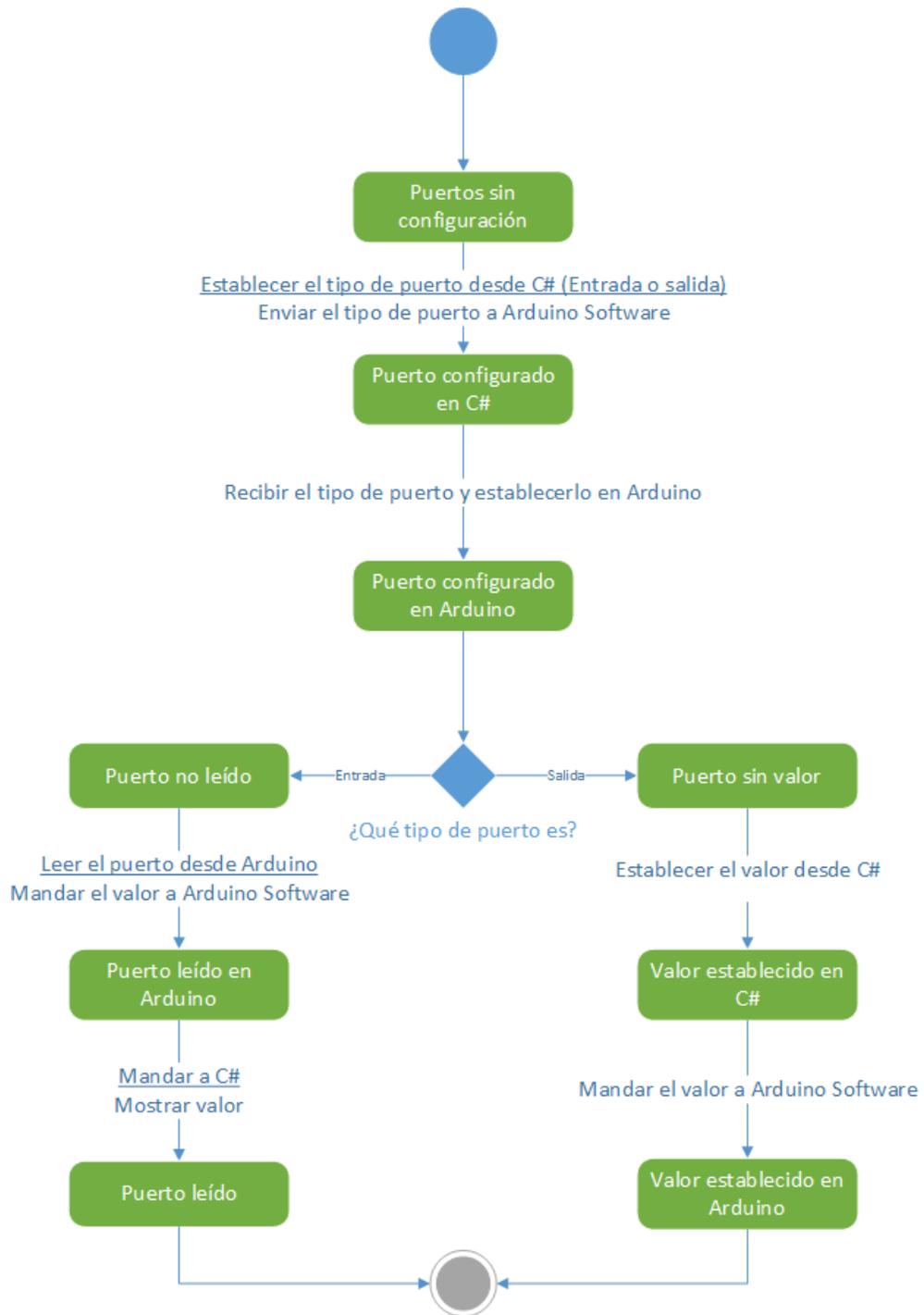


Figura 8. Diagrama de transición de estados para el manejo de puertos (Elaboración propia, 2016).

Capítulo IV. DESARROLLO DEL FRAMEWORK

INTRODUCCIÓN

En este capítulo se presenta el desarrollo del Framework, es decir, la implementación del software de acuerdo la tercera etapa de la metodología en cascada descrita en el capítulo anterior. En esta etapa se ha puesto especial énfasis en la legibilidad del código y facilidad de uso, ya que ésta última es uno de los objetivos de este trabajo.

CONEXIÓN DE ARDUINO CON C# USANDO PUERTO SERIE (USB)

El primer paso consiste en determinar la forma en la cual se hará la comunicación entre Arduino y la PC. El diagrama 1 muestra la forma en que se comunicarán C# y Arduino, el intermediario es el software de Arduino, en el cual, se establecerán métodos para intercambio de datos.



Figura 9.- Comunicación entre C# y Arduino. (Elaboración propia, 2018)

La comunicación es bidireccional, es decir, de C# a Arduino y viceversa. El código de Arduino debe ser estático, no debe ser necesario modificarlo para uso común. Una vez que se ha definido esto, se puede comenzar con el desarrollo. A continuación, se describe de forma resumida el procedimiento que se ha llevado a cabo para esta etapa del modelo de desarrollo de software.

ARDUINO SOFTWARE Y C#

Al comenzar el programa que controlará Arduino se nota que Arduino Software tiene una estructura preestablecida en el código. La documentación del compilador indica que el método `void setup()` debe ser utilizado para las instrucciones que se ejecutan una sola vez, ya que este método se inicia al cargar el programa por única ocasión. El otro método es `void loop()`, el cual, se diseñó para las operaciones repetitivas como se muestra en los comentarios del siguiente código, el cual corresponde a un programa base de Arduino Software.

```
void setup() {  
    // put your setup code here, to run once:  
  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
  
}
```

Código 1.- Estructura básica de un programa en Arduino Software. (Elaboración propia, 2018)

El método `setup()` es usado para configurar puertos e iniciar comunicación por medio del puerto serial, esto según la documentación del compilador. Para poder realizar cualquier configuración de Arduino desde C# se requiere que el primero pueda recibir datos y el segundo enviarlos, por lo tanto, es necesario establecer una comunicación fija antes de poder realizar cualquier otra configuración.

```
1. void setup() {  
    // put your setup code here, to run once:  
    //Configurar tasa de transferencia de datos por usb e iniciar  
    comunicación  
    Serial.begin(9600);  
}
```

Código 2.- Comunicación serial desde Arduino. (Elaboración propia, 2018)

El código anterior inicia la comunicación serial de la placa, no es posible cambiar la tasa de baudios desde C# antes de iniciar la comunicación ya que no se podría recibir la información. Se ha optado por una tasa fija de transferencia de datos de 9600 por ser la más común.

En el método `loop()` se establece la conexión y se procesa la información de la forma más básica posible, con la finalidad de evitar la sobrecarga de procesos en el microcontrolador y aprovechar el microprocesador de la computadora para realizar operaciones complejas, de esta forma se optimiza la ejecución de las aplicaciones.

```
1. void loop() {  
  while (Serial.available()){  
    int instruction=0, port=0, value=0;  
    String data="";  
    data=Serial.readString();  
    instruction=data.substring(0,2).toInt();  
    port=data.substring(2,4).toInt();  
    value=data.substring(4,6).toInt();  
  }  
}
```

Código 3.- Intercambio de datos desde Arduino. (Elaboración propia, 2018)

En el código 3 `Serial.available()` se encarga de vigilar la entrada de datos. Cuando se encuentran datos de entrada `Serial.readString()` los lee y los asigna a una variable de tipo cadena de caracteres que posteriormente será separada en tres partes, la primera será el número de instrucción, la segunda representa en número de puerto y la tercera el valor de los parámetros. La tabla 1 muestra la separación de datos para la cadena recibida “XXYYZZ”.

| XX | YY | ZZ |
|-----------------------|-----------|------------|
| Número de instrucción | Puerto | Parámetros |

Tabla 1.- Interpretación de instrucciones en Arduino. (Elaboración propia, 2018)

En cuanto a C# se utiliza la biblioteca `System.IO.Ports` para el manejo de puertos y se usa la clase `SerialPort`, el constructor de la clase se puede observar en el código 3, recibe un parámetro de tipo string llamada “puerto”, el signo igual asigna un valor en caso de omisión, esto es, se puede llamar de la forma `ArduinoUno()` sin incluir el parámetro.

```
SerialPort serial;  
  
public ArduinoUno(string puerto="COM3") {  
    serial = new SerialPort();  
    serial.PortName = puerto;  
    serial.BaudRate = 9600;  
}
```

Código 4.- Comunicación serial desde C#. (Elaboración propia, 2018)

Se utiliza una tasa de baudios de 9600 para coincidir con la configuración de Arduino, sin embargo, este valor puede ser modificado para tener velocidades de transferencia más altas, recordando que debe coincidir la PC, la configuración de Arduino y C#. Con esto se puede iniciar la comunicación, el resto de la construcción se omite, algunas partes son descritas en la siguiente sección.

Capítulo V. RESULTADOS

Como resultado de la investigación realizada y del desarrollo del Framework mostrado en el capítulo anterior se obtiene una clase en C# llamada `ArduinoUNO.cs` que contiene los métodos realizar operaciones de lectura y escritura de puertos y el uso de sensores de forma eficaz. Además, se creó una rutina en Arduino software para atender las llamadas de C#. A continuación, se describe la implementación y uso del Framework.

USO DEL FRAMEWORK

Para utilizar este Framework es necesario cargar el código que se encuentra en el archivo `conexion.ino` a la tarjeta de Arduino Uno por medio del IDE de Arduino. Para cargar el código, se debe abrir el archivo `conexion.ino` con el IDE de Arduino, dirigirse al menú programa y seleccionar Subir, como se muestra en la imagen 3. Esta acción compila el programa y lo carga a Arduino. El icono  tiene la misma funcionalidad. A este tipo de archivo se le conoce comúnmente como firmware.

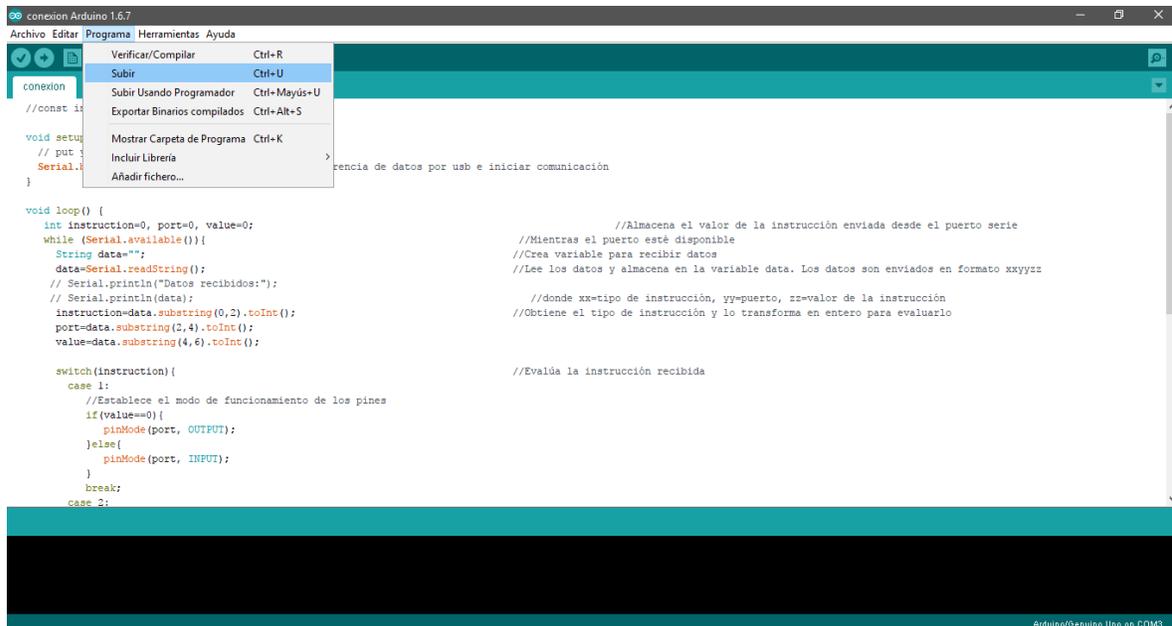


Imagen 3.- Cargar programa a Arduino. (Elaboración propia, 2018)

Con lo anterior Arduino ya contiene los procedimientos necesarios para establecer comunicación con una computadora, lo siguiente es crear un proyecto en C# y agregar la clase que contiene los métodos para el manejo de la placa.

Crear proyecto

Abrir Visual Studio (el Framework ha sido probado en Visual Studio 2017 Community) y crear un nuevo proyecto. En la imagen 4 se ha creado un proyecto de tipo WPF (Windows Presentation Foundation), que es una tecnología que toma características de aplicaciones Windows y aplicaciones Web. También se puede utilizar en otros tipos de proyectos, como Windows Form, Consola, Web o Aplicaciones Universales de Windows, lo que hace a este Framework y en general a C# muy versátil.

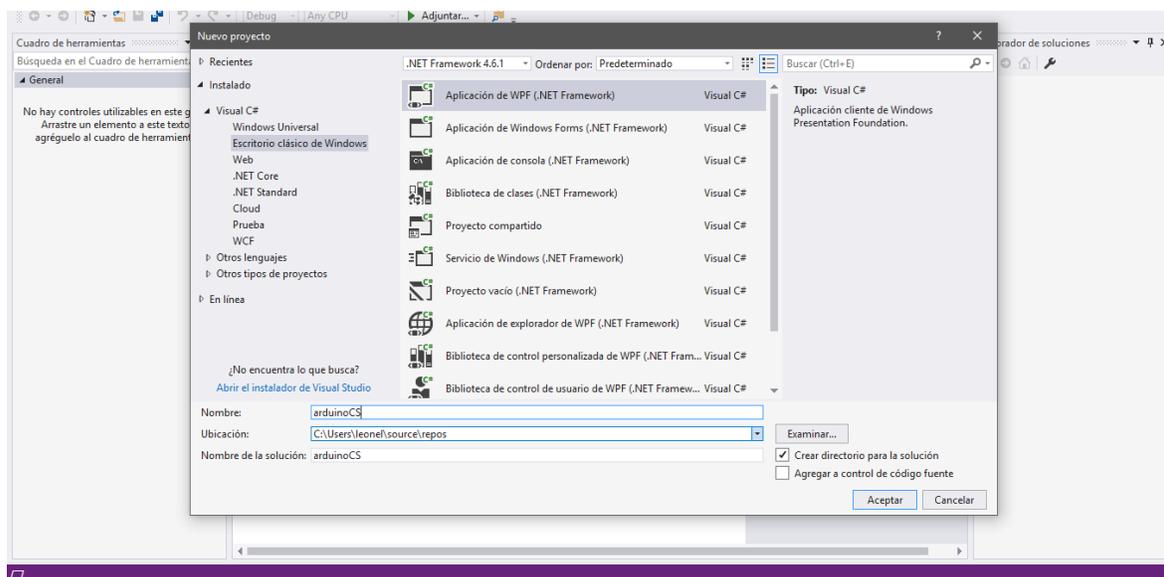


Imagen 4.- Nuevo proyecto en Visual Studio. (Elaboración propia, 2018)

A continuación, se debe asignar un nombre al proyecto y la ubicación en la cual se guardará. Una vez que se ha creado se asignará la clase ArduinoUNO.cs, la cual, contiene los métodos necesarios para trabajar con Arduino. Para cargar la clase, se debe hacer clic derecho sobre el nombre del proyecto en el explorador de soluciones y a continuación, seleccionar Agregar y Agregar elemento existente, como se muestra en la imagen 5.

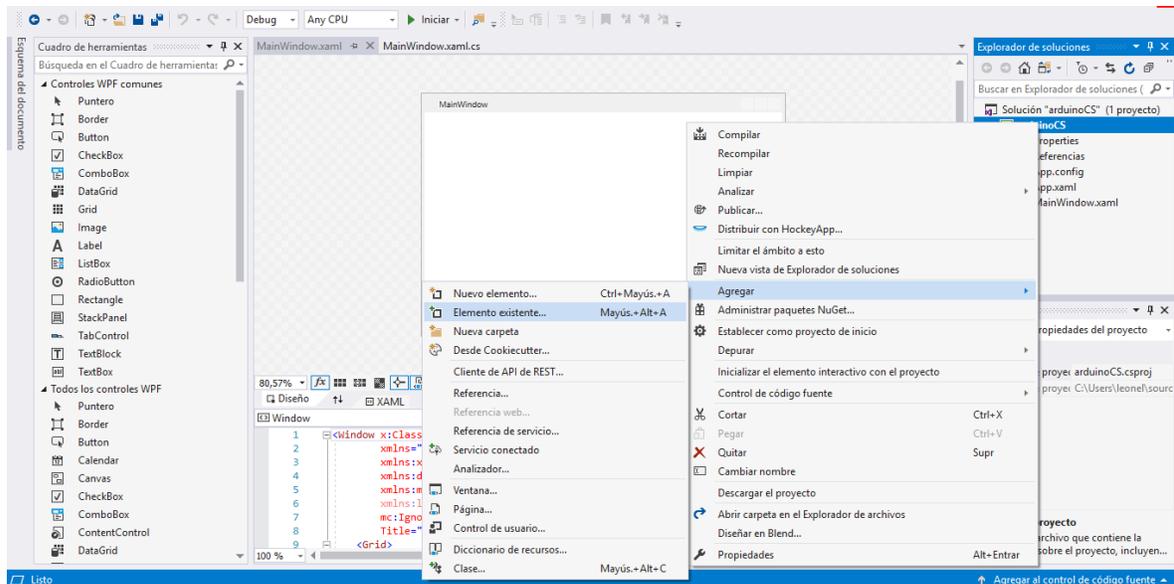


Imagen 5.- Agregar la clase ArduinoUNO. (Elaboración propia, 2018)

Implementación del Framework

Al terminar el procedimiento anterior ya se puede hacer uso del framework, a continuación, se describe la utilización y algunas funciones que contiene. El primer paso es implementar la clase que se cargó anteriormente, para ello podemos agregar la siguiente línea:

```
ArduinoFramework.ArduinoUno arduino=new ArduinoFramework.ArduinoUno();
```

Código 5.- Implementación de la clase ArduinoUNO en C#.

Observe que se ha tomado el nombre de la clase ArduinoFramework y se crea un nuevo objeto llamado arduino usando el constructor vacío, por lo tanto, se tomará el puerto serial COM3. Si se requiere cambiar el nombre del puerto, se debe modificar la línea anterior con el nombre del puerto entre comillas, ya que el parámetro es de tipo String. La siguiente línea muestra el uso del constructor con parámetro.

```
ArduinoFramework.ArduinoUno arduino=new  
ArduinoFramework.ArduinoUno("COM2");
```

Código 6.- Implementación de la clase ArduinoUNO con parámetro en C#. (Elaboración propia, 2018)

En el código 6 se implementó la clase usando el puerto serial COM2, por lo tanto, la placa Arduino debe estar conectado a este puerto. El nombre del puerto puede ser visto en el Administrador de dispositivos de Windows al conectar la placa Arduino, como se ve en la imagen 6. La imagen 7 muestra la implementación de la clase en el proyecto que se creó en la etapa anterior en Visual Studio.

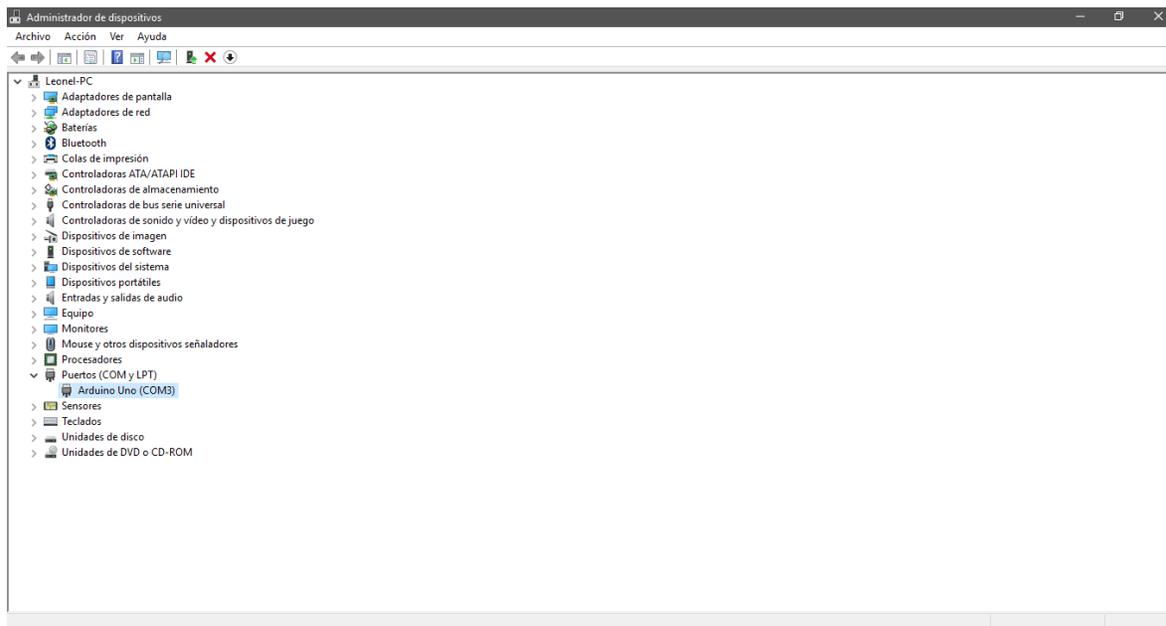


Imagen 6.- Administrador de dispositivos de Windows. (Elaboración propia, 2018)

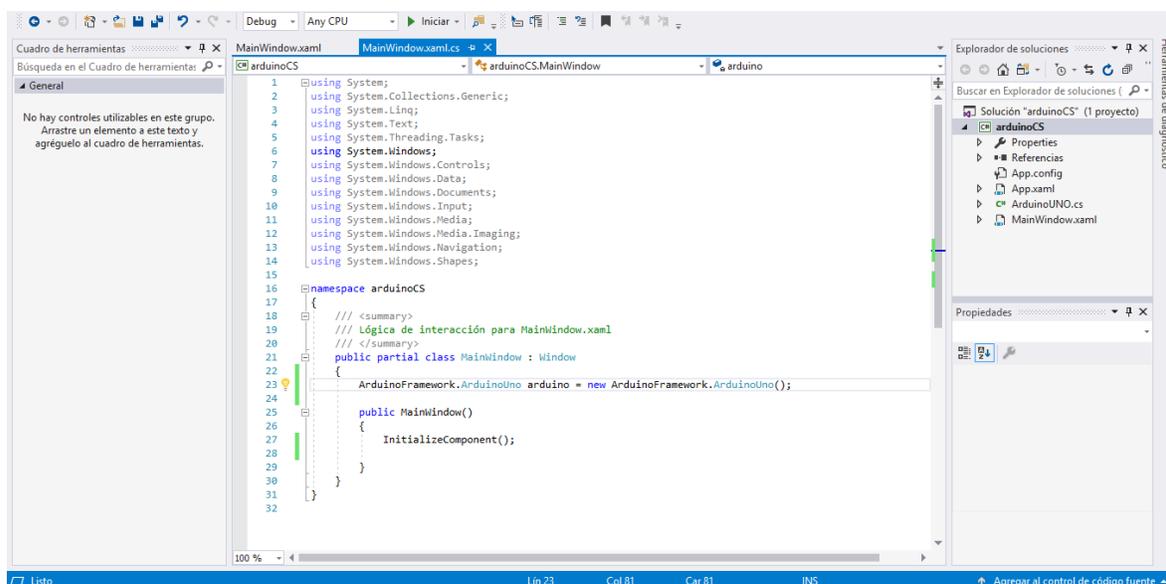


Imagen 7.- Implementación del Framework en Visual Studio. (Elaboración propia, 2018)

Uso del Framework

Ahora se creará un pequeño formulario con un botón en Visual Studio para mostrar la facilidad de uso del Framework y la reducción de líneas de código fuente en la elaboración de aplicaciones. En la imagen 8 se ha arrastrado un botón de los controles WPF comunes al panel.

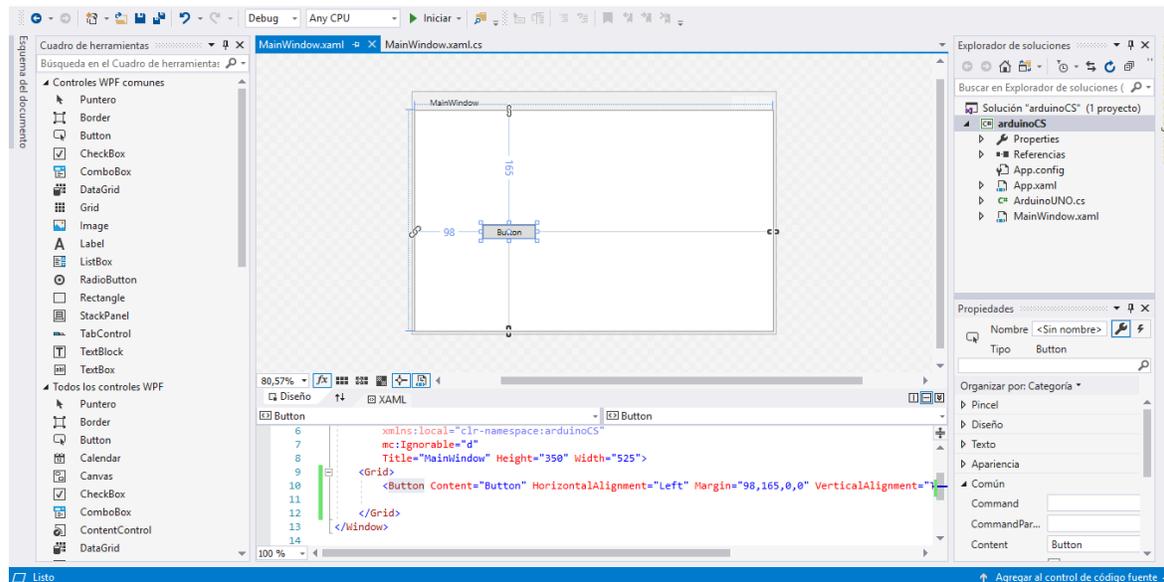


Imagen 8.-Agregar botón en Visual Studio. (Elaboración propia, 2018)

Este botón tendrá la funcionalidad de encender un led en Arduino, para lograr esto, únicamente se debe hacer doble clic sobre el diseño del botón para crear el evento y a continuación, agregar la siguiente línea dentro del método `Button_Click()`.

```
arduino.EscribirDigital(5,1);
```

Código 7.- Método `EscribirDigital()`. (Elaboración propia, 2018)

En el código 7 se muestra el método `EscribirDigital(int pin, int estado)` de este Framework, este método es el equivalente en C# a `digitalWrite` de Arduino. `EscribirDigital` recibe dos parámetros de tipo entero, el primero es para indicar el número de puerto digital y el segundo para el estado que tendrá, es decir, 0 para apagado y 1 para encendido.

Método EscribirDigital(int pin, int estado)

EscribirDigital se puede utilizar en los 14 puertos digitales multipropósito de Arduino, sin embargo, se recomienda no utilizar los puertos 0 y 1, ya que son los encargados de la transmisión de datos por USB.

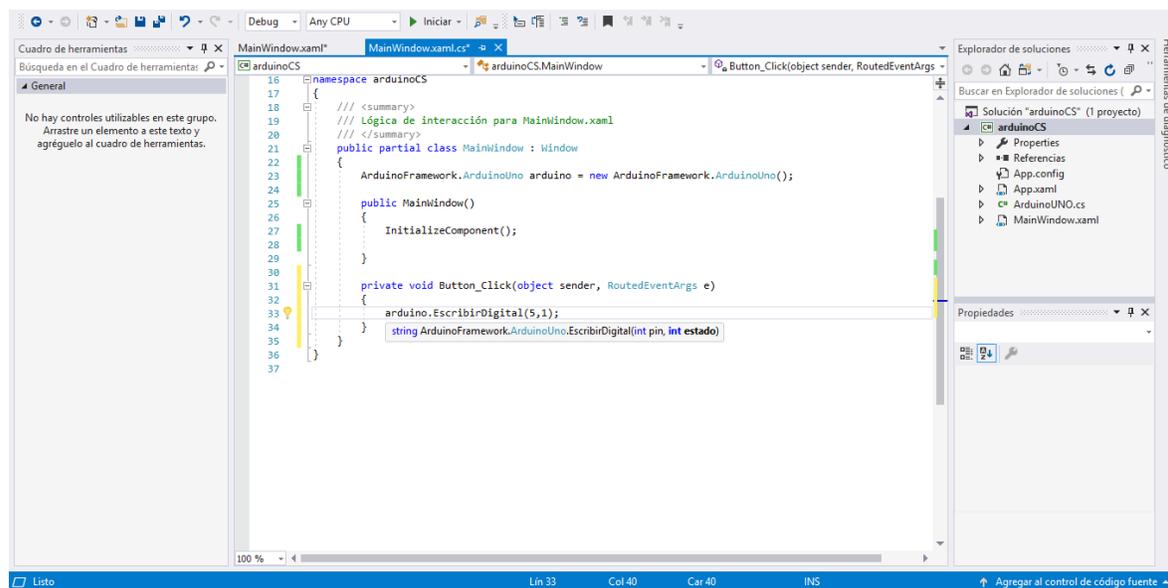


Imagen 9.-Método EscribirDigital en Visual Studio. (Elaboración propia, 2018)

En la imagen 9 se muestra la implementación del método EscribirDigital en Visual Studio, este método también puede utilizarse en los pines analógicos A0-A5, para ello, se debe utilizar la siguiente tabla de equivalencia.

| | |
|----|----|
| A0 | 14 |
| A1 | 15 |
| A2 | 16 |
| A3 | 17 |
| A4 | 18 |
| A5 | 19 |

Tabla 2.- Equivalencia de pines analógicos en el método EscribirDigital. (Elaboración propia, 2018)

Utilizando la tabla anterior se puede usar el pin A0 con el método `EscribirDigital()` asignando el número de pin equivalente 14 como el parámetro `pin`. El código 8 muestra el ejemplo con el pin 14 equivalente a A0.

```
arduino.EscribirDigital(14,1);
```

Código 8.- Equivalencia de pin analógico en el método `EscribirDigital()`. (Elaboración propia, 2018)

El método `EscribirDigital()` es de tipo `String`, devuelve un mensaje de acuerdo con los resultados de su ejecución, los mensajes son:

No se pudo abrir el puerto. – Si el puerto está ocupado, no se cuenta con los privilegios para su utilización o no existe.

No se pudo enviar. – Si ocurre un problema al enviar la información a través del puerto serial.

Se envió correctamente. – Si la información se ha enviado a Arduino.

Método `ConfigurarPuerto(int pin, int tipo)`

Si se ejecuta la aplicación en este punto y se presiona el botón que se creó, no se encenderá el led, esto sucede debido a que no se ha configurado el pin. El método `ConfigurarPuerto(int pin, int tipo)` tiene como finalidad especificar si será un pin de entrada o salida, este método recibe dos parámetros de tipo entero, el primero especifica el número de pin y el segundo el tipo, es decir 0 para entrada y 1 para salida. El código 9 muestra el uso del método `ConfigurarPuerto` con el pin 5 como pin de salida.

```
arduino.ConfigurarPuerto(5,0);
```

Código 9.- Método `ConfigurarPuerto()`. (Elaboración propia, 2018)

El método `ConfigurarPuerto` es de tipo `String`, devuelve los mismos mensajes del método `EscribirDigital`, debe ser utilizado antes de los otros métodos, de preferencia al iniciar la aplicación para tener una buena funcionalidad. En la imagen 10 se ha implementado el método `ConfigurarPuerto` en el constructor de la clase principal, antes de inicializar los componentes, esto permite configurar el pin al iniciar la aplicación y mejorar el rendimiento.

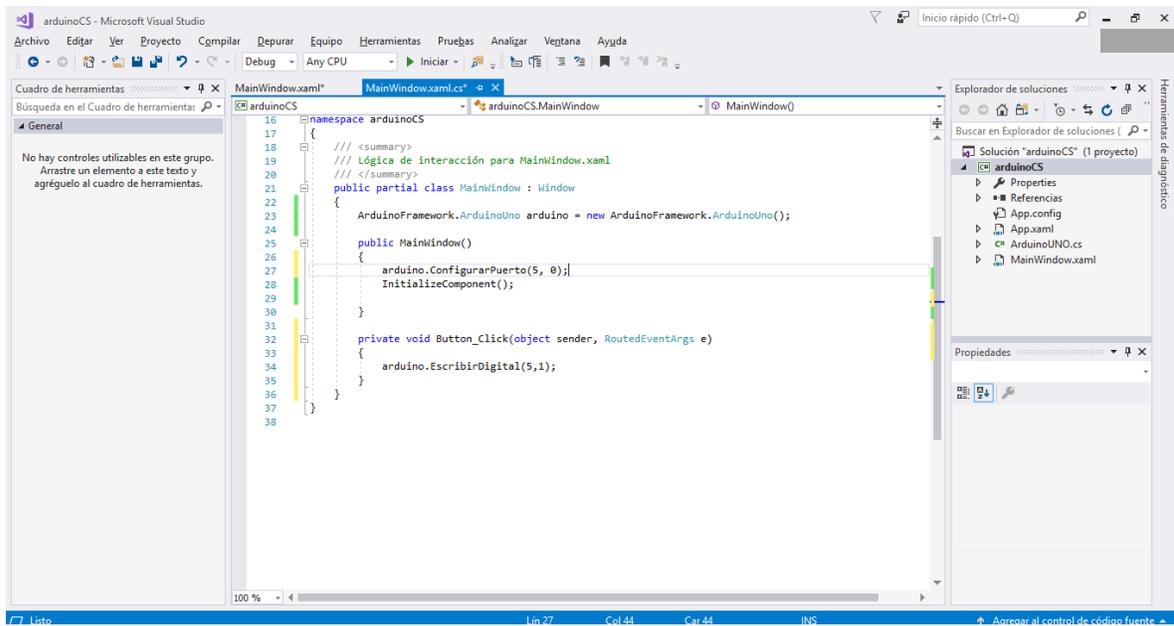


Imagen 10.-Método ConfigurarPuerto en Visual Studio. (Elaboración propia, 2018)

En la imagen 10 se puede apreciar el código fuente final para encender un led, todo se ha reducido a 3 líneas. La siguiente imagen muestra el diagrama de las conexiones de arduino para este ejemplo en el simulador de Tinkercad.

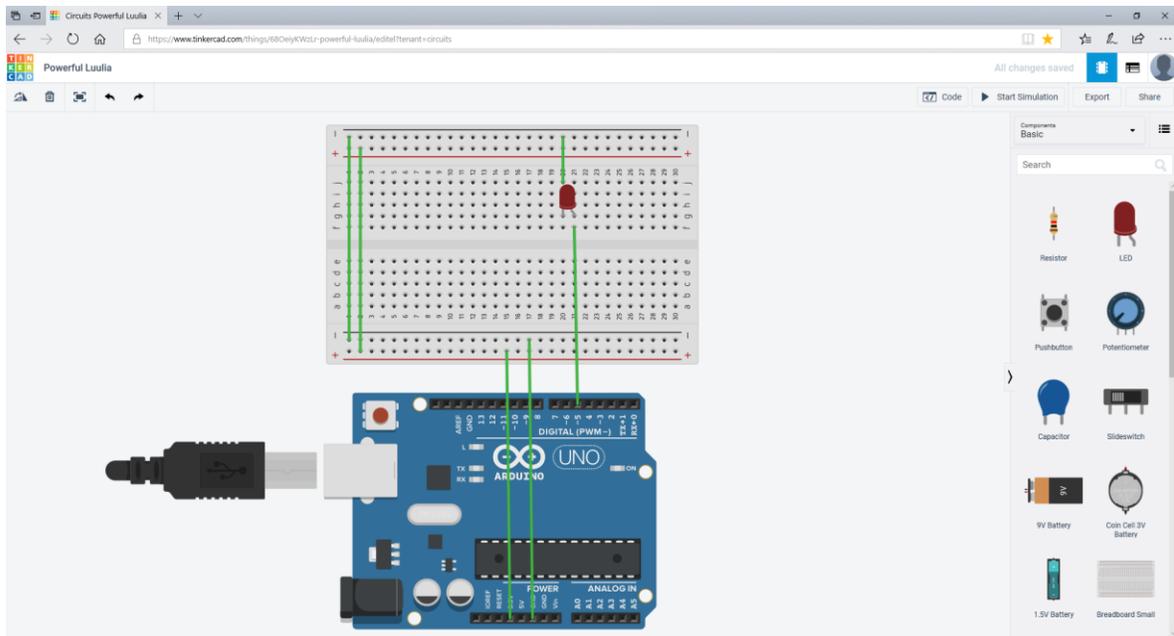


Imagen 11.- Conexiones de Arduino. (Elaboración propia, 2018)

Método EscribirAnalogico(int pin, int estado)

Arduino Uno no posee salidas analógicas puras, pero compensa este punto con salidas PWM, Modulación por ancho de pulso (siglas en inglés de Pulse Width Modulation). Este tipo de modulación consiste en modificar el ciclo de trabajo de una señal periódica senoidal o cuadrada con la finalidad de controlar la cantidad de energía que se envía. En la siguiente imagen tomada de la documentación de Arduino se puede apreciar la modulación por ancho de pulso de manera gráfica.

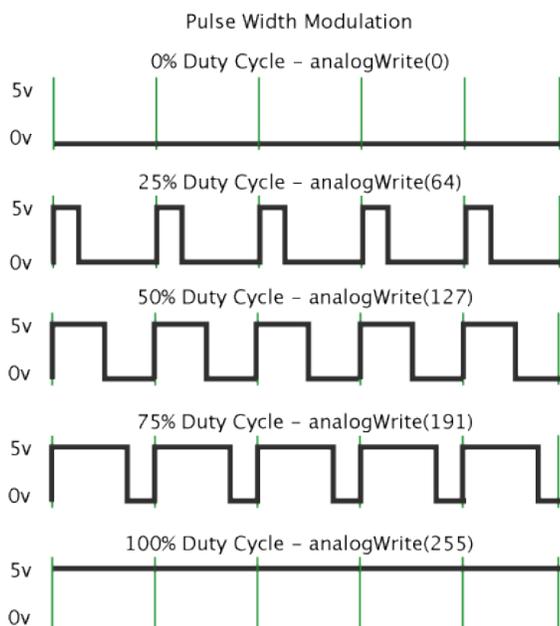


Imagen 12.- Modulación por ancho de pulso. (Arduino, 2018)

Con esta técnica es posible variar el voltaje de salida de un pin entre 0 y 5 volts y así controlar la intensidad de un led o el número de revoluciones de un motor. Arduino cuenta con 6 pines de tipo PWM, los cuales son: 3,5,6,9,10,11.

El método de C# EscribirAnalogico(int pin, int estado) sólo se puede utilizar en pines de tipo PWM, es de tipo String, que devuelve los mismos mensajes de los métodos anteriores y recibe como parámetros el número de pin y el valor a enviar entre 0 y 255. El voltaje que recibe el dispositivo conectado al pin se puede calcular con una equivalencia de 0 a 5 V.

Método LeerAnalogico(int pin)

El método LeerAnalogico(int pin) recibe como parámetro únicamente el número de pin analógico del cual se desea conocer su valor. Arduino Uno cuenta con 6 entradas analógicas numeradas como A0-A5. Este método es de tipo entero y devuelve un valor entre 0 y 1023 que es equivalente a un voltaje de entrada entre 0 y 5 volts.

Los pines de entrada analógica que se utilizan como tal, es decir, como entradas analógicas no requieren de configuración anticipada, esto es, no es necesario utilizar el método ConfigurarPuerto() si el pin se usará como entrada analógica.

Nótese que el parámetro que recibe este método es de tipo entero, por lo tanto, no acepta valores con el nombre del pin como LeerAnalogico(A1) o LeerAnalogico("A1"), en su lugar debe usarse el únicamente el número de la siguiente manera:

```
arduino.LeerAnalogico(0);  
arduino.LeerAnalogico(5);
```

Código 10.- Método LeerAnalogico(). (Elaboración propia, 2018)

En el código 10 se muestra la utilización del método LeerAnalogico de C# con enteros como parámetros de entrada que representan el número de pin analógico. Este método puede utilizarse para obtener valores de un potenciómetro o un sensor. Si se obtiene el valor de un sensor debe ser adaptado a su equivalencia según su hoja de especificaciones.

En la imagen 13 se muestra un diseño del circuito electrónico que se puede usar con los métodos EscribirAnalogico y LeerAnalogico, usando como entrada un sensor de temperatura y un led cuya intensidad depende de la temperatura que detecta el sensor. En la imagen se observa que el led está conectado al pin 3, que puede ser utilizado con PWM.

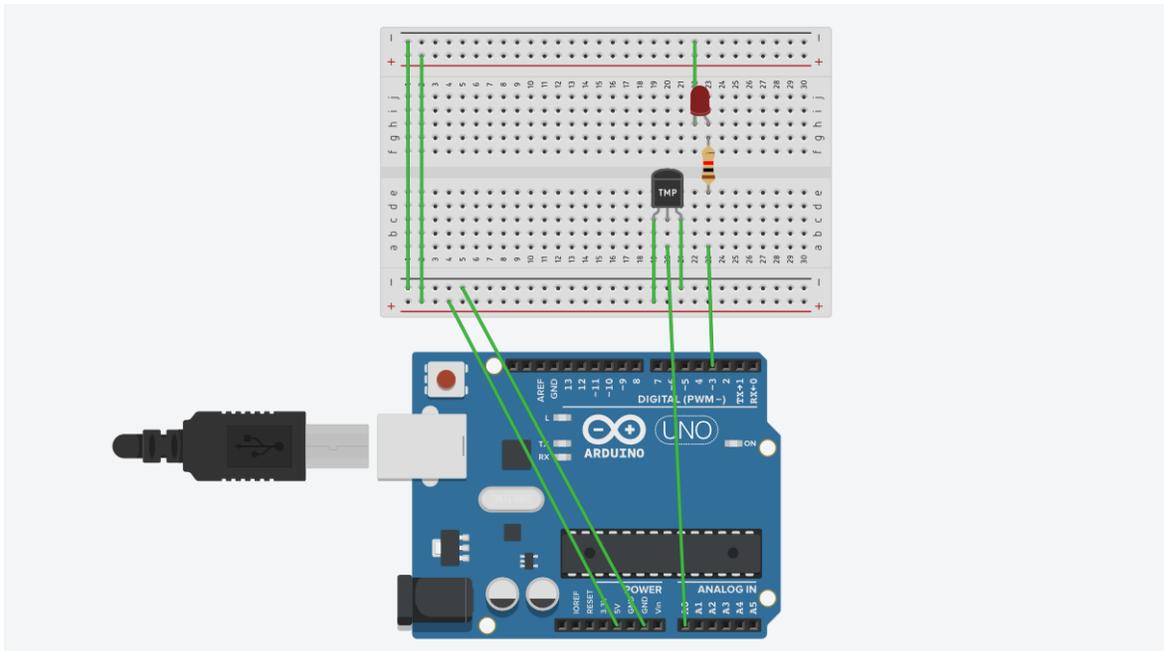


Imagen 13.- Diseño electrónico para el uso de LeerAnalogico y EscribirAnalogico. (Elaboración propia, 2018)

En el siguiente diseño se ha agregado un motor conectado al pin 3 en lugar de un led. Este motor puede ser un pequeño ventilador que gira con una velocidad proporcional a la temperatura recibida del sensor.

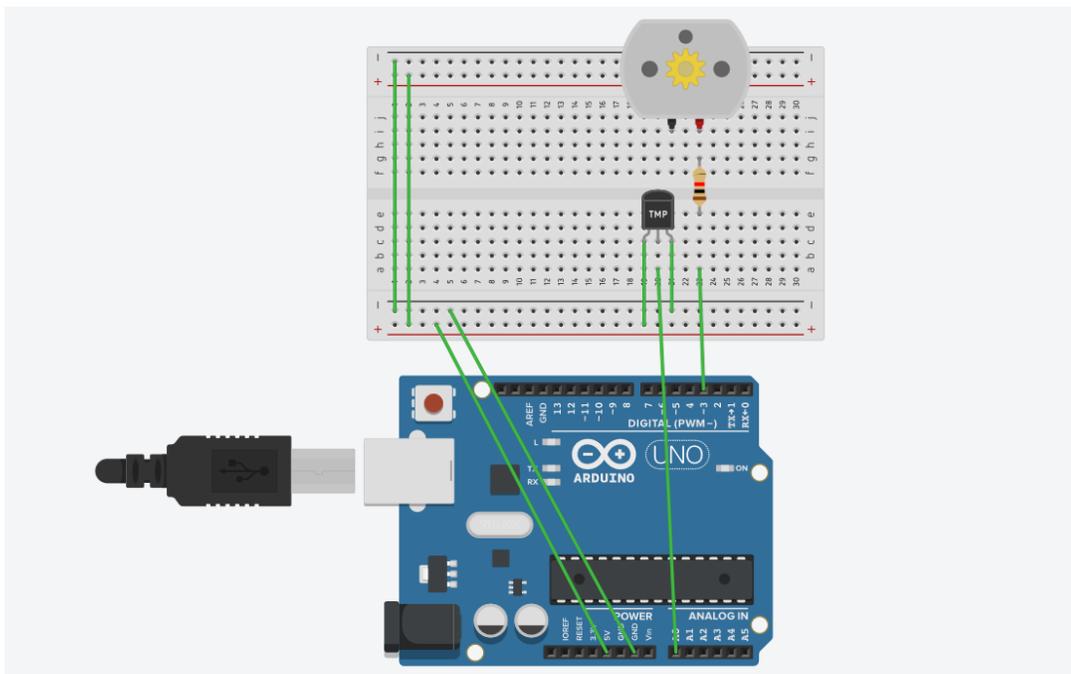


Imagen 14.- Diseño electrónico con motor y sensor de temperatura. (Elaboración propia, 2018)

En el código 11 se asigna de manera directa el valor del sensor de temperatura al led, mientras que en el 12 se ha creado una variable para almacenar el valor de entrada del pin 0. En ambos ejemplos se ha omitido la diferencia que existe entre los valores de entrada y salida, ya que, como se explicó anteriormente el valor de entrada es de 0 a 1023 mientras que el de salida es de 0 a 255. El código 13 muestra la forma de asignar el valor de manera correcta.

```
arduinoEscribirAnalogico.(3,arduino.LeerAnalogico(0));
```

Código 11.- Asignar valor de sensor a led. (Elaboración propia, 2018)

```
int sensor = arduino.LeerAnalogico(0);  
arduinoEscribirAnalogico.(3,sensor);
```

Código 12.- Asignar valor de sensor a motor. (Elaboración propia, 2018)

```
int sensor = arduino.LeerAnalogico(0)/(1023/255);  
arduinoEscribirAnalogico.(3,sensor);
```

Código 13.- Asignar valor equivalente. (Elaboración propia, 2018)

Método Retardo(int milisegundos)

Este método es equivalente a delay() de Arduino, recibe como parámetro un entero que representa la pausa en milisegundos. Retardo() hace una pausa en la ejecución del código C# por el tiempo expresado en su parámetro de entrada. El código 14 muestra su uso para encender y apagar un led después de 1 segundo.

```
arduino.EscribirDigital(14,1);  
arduino.Retardo(1000);  
arduino.EscribirDigital(14,0);
```

Código 14.- Uso del método Retardo(). (Elaboración propia, 2018)

Método LM35(int pin)

El método LM35(int pin) ha sido creado para utilizar el sensor de temperatura LM35, el cual, es fácil de conectar, tiene un bajo costo y existe mucha información sobre este dispositivo. En la siguiente imagen se muestran las terminales de conexión del LM35.

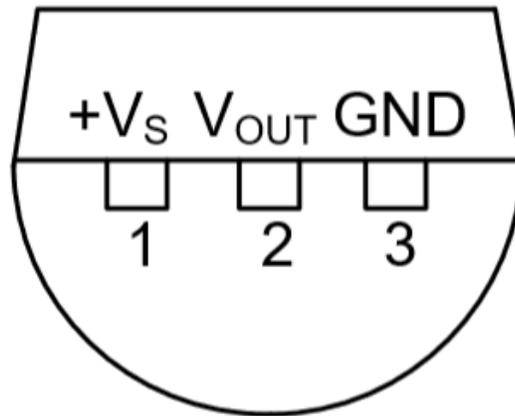


Imagen 15.- Sensor LM35. (Texas Instrument, 2017)

El LM35 requiere un voltaje de entrada de 4 a 20 volts para su funcionamiento en la terminal 1. El voltaje de salida es de entre -550 a 1500 mV que se pueden interpretar de manera directa como -55 a 150 °C, este valor se obtiene del conector 2. La terminal 3 se conecta a GND.

En la imagen 15 se muestra el diseño de conexiones para el uso del sensor con el framework de C#. El valor del sensor se obtiene con el método LM35(0), ya que se encuentra conectado en el pin analógico A0.

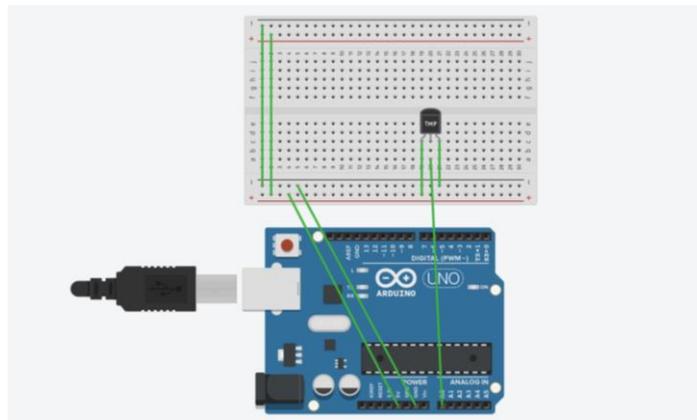


Imagen 16.- Conexión del Sensor LM35 a Arduino. (Elaboración propia, 2018)

Método HCSR04(int trig, int echo)

El método HCSR04 de C# obtiene el valor de un sensor ultrasónico HC-SR04 expresado en centímetros enteros entre 2 y 400. En la imagen 16 se puede apreciar este sensor con las siguientes conexiones:

- Vcc. – Voltaje de alimentación de 5 volts.
- Trig. – Envío del pulso.
- Echo. – Recepción del pulso.
- GND. – Conexión a tierra.



Imagen 17.- Sensor ultrasónico HC-SR04. (Microkits electrónica, 2017)

El método HCSR04() recibe dos parámetros de tipo entero, el primero *trig* representa el pin al que está conectado y que se usará para alimentar el pulso, por lo tanto, debe estar configurado como pin de salida, el segundo parámetro, el *echo* hace referencia al pin en el cual se encuentra conectado, debe ser la entrada analógica que recibe los datos de la medición. En el código 15 se muestra la utilización de este método.

```
int distancia;  
distancia=arduino.HCSR04(2,0);
```

Código 15.- Uso del método HCSR04 de C#

En el ejemplo anterior el conector *trig* debe estar en el pin digital 2 y el *echo* en el pin analógico A0. Los métodos HCSR04 y LM35 se pueden utilizar con hilos paralelos en C# para que permanezcan trabajando durante la ejecución de la aplicación.

Resumen de métodos

En el ejemplo anterior se muestra la implementación y uso del framework, que es totalmente funcional y cuenta con métodos para controlar la placa de Arduino sin necesidad de modificar el firmware. El framework cuenta con los siguientes métodos, además es posible crear más métodos para sensores a partir de LeerDigital() para sensores digitales y LeerAnalogico() para sensores analógicos.

ConfigurarPuerto(int pin, int tipo)

Establece el tipo de uso del puerto.

EscribirDigital(int pin, int estado)

Envía datos de salida a los puertos digitales

EscribirAnalogico(int pin, int estado)

Envía datos de salida a los puertos digitales PWM

LeerDigital(int pin)

Recibe el valor de un puerto digital

LeerAnalogico(int pin)

Recibe el valor de un puerto analógico

Retardo(int tiempo)

Realiza una pausa en para sincronizar Arduino

LM35(int pin)

Lee el valor de un sensor de temperatura LM35

HCSR04(int trig, int echo)

Lee el valor de un sensor ultrasónico HC-SR04

CONCLUSIONES Y TRABAJO FUTURO

Con la realización de este trabajo de investigación se comprueba que:

- Es posible simplificar la programación de aplicaciones realizadas con C# para el control de Arduino utilizando el framework propuesto.
- El uso de los métodos reduce considerablemente el número de líneas de código y por lo tanto el tiempo de desarrollo, esto puede comprobarse en el Anexo 1.
- Puede manipularse una placa Arduino desde un programa realizado en C# sin utilizar Arduino Software cuando la placa contiene el código del firmware desarrollado en este trabajo.
- No es necesario conocer el lenguaje de Arduino para utilizar este framework.
- Los desarrolladores que tienen conocimientos sobre el uso de Arduino IDE y C# encontrarán una herramienta que puede ser adaptado a sus trabajos o ampliado por ellos mismos de forma sencilla.
- Una clase es suficiente para poder controlar Arduino desde C#, cuando la placa contiene el firmware del framework.

El principal aporte generado en la investigación es la reducción del tiempo de desarrollo en aplicaciones que involucre el uso de Arduino y una PC, lo cual, se espera que motive a estudiantes, profesionales de la computación, electrónica, mecatrónica y demás áreas afines a crear más y mejores aplicaciones.

En México es necesario fomentar la creación de programas que permitan el desarrollo del internet de las cosas, este trabajo representa un inicio para todos aquellos que requieran conocer su estructura básica y los conceptos necesarios para entender su funcionamiento.

Concluido este trabajo de investigación se propone el siguiente listado de trabajo a futuro:

1. Agregar más métodos para otros sensores. A partir de los métodos LeerDigital() y LeerAnalogico() se pueden agregar a C# nuevos sensores de una forma sencilla y estandarizada.
2. Demostrar que el código de Arduino incluido en el firmware se puede utilizar para crear otro framework en otro lenguaje sin modificación. La estructura de los métodos

de lectura y escritura en el búfer es similar en varios lenguajes de alto nivel, por lo que el firmware de Arduino desarrollado en este trabajo se adaptará sin problemas a muchos lenguajes.

3. Crear un framework equivalente en Java. Java es un lenguaje muy similar a C#, por lo que crear un framework con la misma funcionalidad será sencillo y rápido, requerirá de pocas adaptaciones y se podrá usar en varias plataformas.
4. Utilizar el framework de Java en aplicaciones de Android. Una vez que se tenga el framework en Java es posible adaptarlo en aplicaciones de Android, además se puede crear una comunicación por bluetooth o Wifi para utilizarlo en dispositivos inalámbricos.
5. Comparar el rendimiento de las aplicaciones creadas con el framework de Java contra las de C# y Xamarin. Xamarin permite la creación de apps multiplataforma escritas en C#, es decir, se pueden usar en IOS o Android. Una vez creado el framework en Java y aplicado a crear apps en Android, se pueden comparar con las desarrolladas en C# y medir la diferencia en rendimiento. Con esto se puede demostrar que con el framework de C# desarrollado en este trabajo se pueden desarrollar apps multiplataforma con un rendimiento similar al software nativo.

REFERENCIAS

Arduinio. (2018). *Arduino Products*. Recuperado el 10 de enero de 2018, de <https://www.arduino.cc/en/Main/Products>.

Barranco, J. (2001). *Metodología del análisis estructurado de sistemas*. Madrid: Universidad Pontificia Comillas

Bertschi M. (2014). *Arduino, C#, and Serial Interface*. Recuperado el 12 de Noviembre de 2015 de <http://www.codeproject.com/Articles/473828/Arduino-Csharp-and-Serial-Interface>.

Cisco. (2015). *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are*. Recuperado el 4 de septiembre de 2018 de <file:///C:/Users/leo85/Documents/tesis/computing-overview.pdf>

Electrónica PIC. (2015). *Encender y apagar un Led con Arduino y Visual Studio 2015*. Recuperado el 4 de mayo de 2018 de <http://electronica-pic.blogspot.mx/2015/11/encender-y-apagar-un-led-con-arduino-y.html>

Esqueda, J. A. S., López, A. S., Espino, V. R., & Salazar, R. M. (2014). "Seguimiento de Objetos empleando Aforge. Net y Arduino". *Revista Arquitectura e Ingeniería*, 8(1): 2-12.

Forbes. (2018). *CES 2018 | El mundo ya se conectó al Internet de las Cosas y así funciona*. Recuperado el 10 de Enero de 2018 de <https://www.forbes.com.mx/ces-2018-el-mundo-ya-se-conecto-al-internet-de-las-cosas-y-asi-funciona/>

Fowler, M. (2004). *UML Distilled: A Brief Guide to the Standard Object Modeling*. Addison Wesley.

Galindo J. y Campos J. (2008). Diseño e implementación de un marco de trabajo (framework) de presentación para aplicaciones JEE. Recuperado el 11 de Enero de 2018 de <http://openaccess.uoc.edu/webapps/o2/bitstream/10609/876/1/00765tfc.pdf>

García, X. (2014). *Desarrollo y reutilización de componentes software y multimedia mediante lenguajes de guion*. España: Paraninfo S.A.

Kendall, E. (2005). *Análisis y diseño de sistemas*. México: Pearson Educación.

Kurniawan A. (2012). *Getting Started with Arduino and .NET*. Alemania: PE Press.

Laudon, K. y Laudon, J. (2004). *Sistemas de información gerencial*. México:

Mark Hung, Gartner Research (2017). Leading the IoT Recuperado el 13 de diciembre de 2017 de

https://www.gartner.com/imagesrv/books/iot/iotEbook_digital.pdf

Microchip. (2018). *ATmega328*. Recuperado el 10 de Enero de 2018 de <https://www.microchip.com/wwwproducts/en/ATmega328>

Microkits Electrónica. *Sensor ultrasonido*. Recuperado el 4 de septiembre de 2018 de <https://www.microkitselectronica.com/media/attachment/file/h/c/hcsr04.pdf>

Microsoft (2015). *SerialPort Class*. Recuperado el 12 de Noviembre de 2015 de [https://msdn.microsoft.com/en-](https://msdn.microsoft.com/en-us/library/system.io.ports.serialport(v=vs.110).aspx)

[us/library/system.io.ports.serialport\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.io.ports.serialport(v=vs.110).aspx).

Mojica F. (2011). *Controla a tu Arduino con C#, FIRMA_SQCNC y la clase Arduino de C#*. Recuperado el 11 de Noviembre de 2015 de <http://hzsquare.blogspot.mx/2011/06/controla-tu-arduino-con-c-firmasqcnc-y.html>.

NetMarket. (2015). *Desktop Operating System Market Share*. Recuperado el 18 de octubre de 2015, de <http://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=0>

Pérez, R. G. (2011). *Net Zeppelin: dirigible con C# y estación de tierra*. Tesis de licenciatura no publicada. Universidad Politécnica de Cataluña. España.

Pressman, R. (2002). *Ingeniería del Software. Un enfoque práctico*. España: Mc Graw Hill 5ª edición.

Price J. & Gunderloy M. (2002). *Mastering Visual C# .NET*. EE. UU: SYBEX.

PrimeTek (2015). *Primefaces*. Recuperado el 11 de Noviembre de 2015 de <http://www.primefaces.org/>.

proceso para el desarrollo de Software. México: Cengage Learning.

Reyna G., Soriano J., Miranda M., Herrera, J., Taud, H., & Álvarez J. *Arduino uno para prototipado rápido: Generador de números aleatorios*. *Contactos*, 91:54–59.

Sarmiento S. (2015, 7 de enero). *Sergio sarmiento / internet de cosas*. El Norte. México.

Sommerville, I. (2005). *Ingeniería del software*. Madrid: Pearson educación. (7ma ed).

Sung, W. T., Chen, J. H., Hsiao, C. L., & Lin, J. S. (2014, Junio). Multi-sensors Data Fusion Based on Arduino Board and XBee Module Technology. In *Computer, Consumer and Control (IS3C), 2014 International Symposium*” 14417418:422-425. IEEE.

Taboada, J. y Cotos, J. (2005). *Sistemas de información medioambiental*. España.

Tamayo, M. (2004). *El proceso de la investigación científica*. México: Limusa.

Texas Instruments. (2017). *LM35 Precision Centigrade Temperature Sensors*. Recuperado el 4 de septiembre de 2018 de <http://www.ti.com/lit/ds/symlink/lm35.pdf>

The Social Intelligence Unit A.C SIU. (2017). *Internet de las Cosas en México*. Recuperado el 10 de Enero de 2018 de http://www.the-siu.net/nwsltr/SIU_75.html

Van Luc. (2014). *jQuery: el framework JavaScript de la Web 2.0*. España: Ediciones ENI.

Vega E., Santamaría P. & Rivas T. (2014). “Internet de los objetos empleando arduino para la gestión eléctrica domiciliaria”. *Revista Escuela de Administración de Negocios*, 77:24-41. Recuperado de <http://www.redalyc.org/articulo.oa?id=20633274004> el 29 de Septiembre de 2015.

Weitzenfeld, A. (2005). *Ingeniería de software: Orientada a objetos con UML, Java e Internet*. Thomson.

Weitzenfeld, A. y Guardati, S. (2007). *Capítulo 12: Ingeniería de Software: el proceso para el desarrollo de Software*. México: Cengage Learning.

Zachariadou, K., Yiasemides, K., & Trougkakos, N. (2012). “A low-cost computer-controlled Arduino-based educational laboratory system for teaching the fundamentals of photovoltaic cells”. *European Journal of Physics*, 33(6): 1599.

Zend. (2015). *Zend framework 2.0*. Recuperado el 11 de Noviembre de 2015 de <http://framework.zend.com/>.

Zapata, C. y Garcés, G. (2008). *Generación del diagrama de secuencias de UML 2.1. 1 desde esquemas preconceptuales*. Revista EIA, pp. 89-103.

ANEXO 1 COMPARACIÓN DE CÓDIGOS

A continuación, se presenta dos programas que se utilizan para mostrar la diferencia en la construcción de una aplicación que requiere del uso de C# y Arduino. El primer código fue tomado de Electrónica PIC (2015) y consiste en encender y apagar un led. El segundo tiene la misma funcionalidad usando el framework que se desarrolla en este trabajo.

CÓDIGO ARDUINO (SIN FRAMEWORK)

```
pinMode(pinLed1, OUTPUT);
char character;
String comando;
void setup() {
    Serial.begin(115200);
}
void loop() {

while (Serial.available() > 0) {
    character= Serial.read();
    comando.concat(character);
    delay(10);
}

if (comando.equals("Led_ON") == true) {
    digitalWrite(pinLed, HIGH); // Enciende el Led.
    Serial.println("Led 13 encendido.");
}
if (comando.equals("Led_OFF") == true) {
    digitalWrite(pinLed, LOW); // Apaga el Led.
    Serial.println("Led 13 apagado.");
}
comando="";
}
}
```

CÓDIGO C# (SIN FRAMEWORK)

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;

namespace Arduino_Led{
    public partial class Form1 : Form{
        public Form1(){
            InitializeComponent();

            if(!serialPort1.IsOpen){
                try{
                    serialPort1.Open();
                }
                catch (System.Exception ex) {
                    MessageBox.Show(ex.ToString());
                }
            }
        }

        private void button_Encender_Click(object sender, EventArgs e)
        {
            byte[] mBuffer = Encoding.ASCII.GetBytes("Led_ON");
            serialPort1.Write(mBuffer, 0, mBuffer.Length);
        }

        private void button_Apagar_Click(object sender, EventArgs e)
        {
            byte[] mBuffer = Encoding.ASCII.GetBytes("Led_OFF");
            serialPort1.Write(mBuffer, 0, mBuffer.Length);
        }
    }
}
```

CÓDIGO ARDUINO (CON FRAMEWORK)

```
/*No se requiere Código ya que se encuentra en el firmware*/
```

CÓDIGO C# (CON FRAMEWORK)

```
ArduinoFramework.ArduinoUno arduino=new ArduinoFramework.ArduinoUno();  
arduino.ConfigurarPuerto(5,0);  
arduino.EscribirDigital(5,1);  
arduino.EscribirDigital(5,0);
```

RESULTADOS

En el primer código se han remarcado con negritas únicamente las líneas que son generadas por el programador y se descartan aquellas que el compilador agrega de forma automática. En el código que usa el framework sólo se muestran las líneas que el programador escribe. Los resultados finales son:

| | Arduino (Sin Framework) | C# (Sin Framework) | Arduino (Con Framework) | C# (Con Framework) |
|-------------------------|--------------------------------|---------------------------|--------------------------------|---------------------------|
| Líneas de código | 19 | 12 | 0 | 4 |

| | Sin Framework | Con Framework |
|-------------------------|----------------------|----------------------|
| Líneas de código | 31 | 4 |

ANEXO 2 DIAGRAMAS IPO

Los diagramas IPO permiten mostrar las entradas, los procesos y las salidas de las aplicaciones y las funciones de estas. A continuación, se presentan los diagramas IPO del framework.

Conectar

| ENTRADA | PROCESO | SALIDA |
|----------------|--|---|
| Puerto | <ol style="list-style-type: none">1. Enviar el puerto desde la clase de C# a la clase de Arduino Software.2. Enviar el puerto de Arduino Software a la placa.3. Establecer conexión.4. Revisar conexión5. Regresar valor de conexión exitosa o fallida | Valor de conexión lista o fallida. |

Configurar puerto

| ENTRADA | PROCESO | SALIDA |
|----------------------|--|---------------------------------------|
| Puerto, valor | <ol style="list-style-type: none">1. Enviar la configuración del puerto desde la clase de C# a la clase de Arduino Software.2. Enviar la configuración del puerto de Arduino Software a la placa.3. Establecer configuración del puerto. | Configuración del puerto lista |

Escribir en el puerto

| ENTRADA | PROCESO | SALIDA |
|----------------------|---|--------------------------------------|
| Puerto, valor | <ol style="list-style-type: none">1. Enviar el valor del puerto desde la clase de C# a la clase de Arduino Software.2. Enviar el valor del puerto de Arduino Software a la placa.3. Establecer el valor del puerto en la placa. | Valor del puerto establecido. |

Leer el puerto

| ENTRADA | PROCESO | SALIDA |
|----------------|--|--------------------------------|
| Puerto | <ol style="list-style-type: none">1. Enviar el puerto a leer desde la clase de C# a la clase de Arduino Software.2. Enviar el puerto a leer de Arduino Software a la placa.3. Obtener el valor del puerto. | Valor del puerto leído. |

Configurar sensor

| ENTRADA | PROCESO | SALIDA |
|----------------|---|--|
| Puerto | <ol style="list-style-type: none">1. Enviar el intervalo de pines que usará el sensor desde la clase de C# a la clase de Arduino Software.2. Enviar intervalo de pines que usará el sensor de Arduino Software a la placa.3. Establecer intervalo de pines que usará el sensor. | Configuración del sensor establecida. |

Leer el sensor

| ENTRADA | PROCESO | SALIDA |
|---------------|--|--------------------------------|
| Puerto | <ol style="list-style-type: none"><li data-bbox="680 296 992 470">1. Enviar el intervalo de pines que usa el sensor desde la clase de C# a la clase de Arduino Software.<li data-bbox="680 478 992 617">2. Enviar el intervalo de pines que usa el sensor de Arduino Software a la placa.<li data-bbox="680 625 992 690">3. Obtener el valor del sensor. | Valor del sensor leído. |

ANEXO 3: DIAGRAMAS DE SECUENCIA

Los diagramas de secuencia muestran la interacción de una parte de la aplicación a través del tiempo. Los siguientes diagramas describen la funcionalidad de algunas partes del framework.

DIAGRAMA DE SECUENCIA PARA LA CONEXIÓN

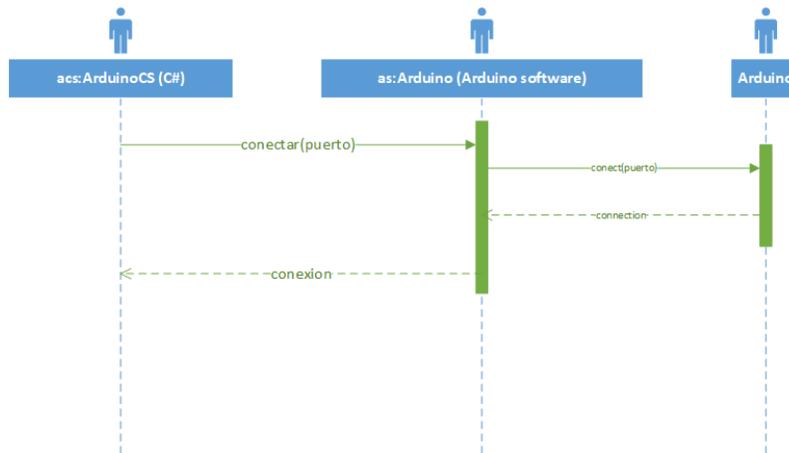
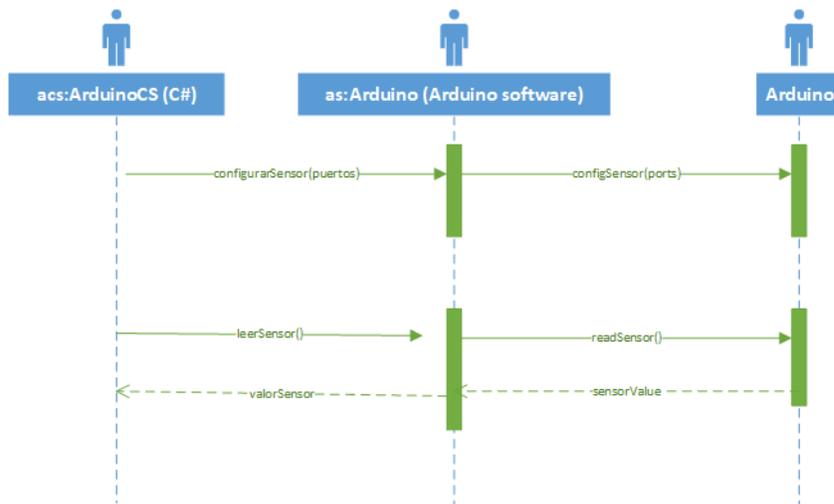


DIAGRAMA DE SECUENCIA PARA EL MANEJO DE SENSORES



ANEXO 4: DIAGRAMAS DE TRANSICIÓN

Este tipo de diagramas son representaciones gráficas que muestran el estado de cada componente en un punto de tiempo diferente, así como los cambios que van ocurriendo conforme avanza su ejecución.

DIAGRAMA DE TRANSICIÓN DE ESTADOS PARA LA CONEXIÓN

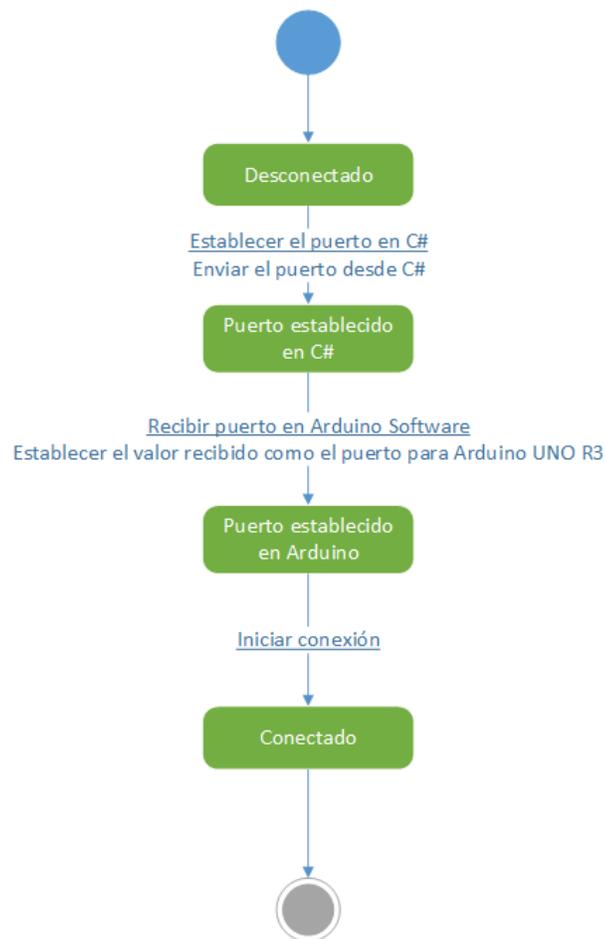


DIAGRAMA DE TRANSICIÓN DE ESTADOS PARA EL MANEJO DE PUERTOS

