INSTITUTO TECNOLÓGICO DE CHIHUAHUA

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

"DESARROLLO DE UN CONTROLADOR DÉBILMENTE ACOPLADO PARA INCUBADORAS NEONATALES"

TESIS

QUE PARA OBTENER EL GRADO DE

MAESTRO EN INGENIERÍA MECATRÓNICA

PRESENTA:

FABIÁN FRANCO LUNA

DIRECTOR(A) DE LA TESIS:

DR. JOSÉ EDUARDO ACOSTA CANO DE LOS RÍOS

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

CHIHUAHUA, CHIH. JULIO DEL 2020

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CHIHUAHUA

"2020, Año de Leona Vicario, Benemérita Madre de la Patria"

Chihuahua, Chih., 30 de junio de 2020

M.C. ROGELIO E. BARAY ARANA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN
PRESENTE

Por medio de la presente notificamos a usted que en cumplimiento de los requerimientos para la obtención de grado de Maestro, el documento de tesis del c. **FABIAN FRANCO LUNA**, ha sido aprobado y aceptado para su impresión. El título de la tesis es:

"DESARROLLO DE UN CONTROLADOR DÉBILMENTE ACOPLADO PARA INCUBADORAS NEONATALES"

Por lo que proponemos, le sea concedida la autorización de impresión correspondiente.

Agradeciendo la atención a la presente, quedamos de usted:

ATENTAMENTE
Excelencia en Educación Tecnológica.

Dr. José E. Acosta Cano de los Ríos
MIEMBRO DEL JURADO DE EXAMEN

Dr. Pedro R. Acosta Cano de los Ríos
MIEMBRO DEL JURADO DE EXAMEN

M.C. Pedro Rafael Márquez Gutiérrez
MIEMBRO DEL JURADO DE EXAMEN

Dr. Oscar Arturo Chavez López
MIEMBRO DEL JURADO DE EXAMEN

EDUCACIÓN | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CHIHUAHUA
E INVESTIGACIÓN

c.c.p.archivo
/reba

EDUCACIÓN

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CHIHUAHUA

"2020, Año de Leona Vicario, Benemérita Madre de la Patria"

Chihuahua, Chih., 30 de junio de 2020

FABIAN FRANCO LUNA
PRESENTE

Por este conducto le comunico que, a propuesta del Jurado de Examen, la División de Estudios de Posgrado e Investigación le ha concedido la autorización para la impresión de tesis para obtener el grado de Maestro, cuyo título es:

"DESARROLLO DE UN CONTROLADOR DÉBILMENTE ACOPLADO PARA INCUBADORAS NEONATALES"

Con el siguiente contenido de capítulos:

I. Introducción.
II. Marco Teórico.
III. Planteamiento del problema.
IV. Desarrollo de la solución.
V. Sistema en operación
VI. Conclusiones.

ATENTAMENTE
Excelencia en Educación Tecnológica.

M.C. ROGELIO [illegible] ARANA
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

EDUCACIÓN
INSTITUTO TECNOLÓGICO DE CHIHUAHUA
DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

c.c.p. archivo
frdm

AGRADECIMIENTOS

Agradecimientos al Tecnológico Nacional de México campus Chihuahua o como para mí y muchos otros “el Tec de Chihuahua” por ser mi hogar durante mi carrera profesional, licenciatura y ahora maestría.

Así mismo, al CONACYT por brindarme su apoyo y permitir dedicar mi tiempo completo a la elaboración de este proyecto.

Fabián Franco Luna

Resumen

El mantenimiento realizado a incubadoras neonatales, en especial el correctivo, representa una inversión importante por parte de las instituciones que las albergan, mayor incluso al considerar el corto tiempo en el que se mantiene dentro de los rangos de operación correctos sin requerir el reemplazo total del equipo ya que generalmente los equipos son diseñados para trabajar con sensores y actuadores específicos y si alguno de los proveedores falla en dar reemplazo al mismo el funcionamiento del equipo no puede continuar y el equipo debe ser desechado.

La presente tesis establece una solución a la incorporación y configuración de elementos que no fueron contemplados al momento de diseño del equipo, permitiendo así la reutilización del controlador desarrollado en plataformas ya establecidas alargando por lo tanto la vida de los equipos. Tomando como base el esquema de referencia ArquiTAM para lograr el acoplamiento débil entre el controlador y los elementos que lo componen (sensores y actuadores) se crea un marco de trabajo para satisfacer las necesidades cambiantes del mercado en cuanto a oferta de transductores que permitan medir y/o controlar las distintas variables del entorno (temperatura, humedad, oxigeno…), así como de la flexibilidad para aumentar la capacidad de funciones que se pueden tener. Los resultados mostrados en el presente trabajo comprueban la eficacia del marco de trabajo creado a partir del esquema ArquiTAM para la incorporación de nuevos elementos al equipo sin necesidad de modificar el código del mismo, y de portabilidad gracias al uso de la plataforma universal de Windows (UWP).

Índice

Listas de figuras

2. Marco Teórico

4. Desarrollo de la solución

5. Resultados

Lista de tablas

2. Marco Teórico

4. Desarrollo de la solución

I. Introducción

La salud es un tema de interés social debido a que ninguna persona se encuentra exenta de padecer una enfermedad o condición que afecte su calidad de vida e incluso atente en contra de la vida misma. Gracias a los grandes avances que se han suscitado a través de la historia se ha dado solución a una gran cantidad de padecimientos a través de tratamientos o cirugías, al igual que la investigación y desarrollo de nuevos medicamentos y equipos que de manera constante han permitido lograr un estado que en la mayoría de los casos padecimientos que en la anterioridad pudieron haber resultado en la muerte del paciente, hoy en día se han solucionado por completo o se han logrado contener de tal modo que la esperanza de vida se ha alargado y se ha logrado salvar la vida de millones de personas.

Tal es el caso de la incubadora neonatal definida como un equipo de soporte de vida destinado a recién nacidos prematuros, con el objetivo de mantener las condiciones ambientales para completar su correcto desarrollo. Desde su invención, se ha disparado la tasa de preservación de los bebés recién nacidos, esto igualmente dado por la mejora de condiciones ambientales controladas y de salubridad, así como las técnicas y equipos de apoyo para el parto.

Continuamente se buscan nuevas maneras de mejorar su desempeño e incluir nuevas funciones con el objetivo de reducir la tasa de defunciones e incluso tratar padecimientos más complejos y ayudar en periodos mayores al del desarrollo del infante. Actualmente siguen presentándose trabajos que pretenden la incorporación de acciones de control poco convencionales e incluso en mejoras acerca de su constitución física. Generalmente este tipo de equipos cuentan con un funcionamiento básico y limitado en cuanto a funciones se refiere. Más aun, la compatibilidad entre sensores y actuadores, y soporte proporcionado a este tipo de equipos es muy limitado por el fabricante, lo cual hace que estos equipos tengan una vida útil corta. El objetivo de este trabajo de tesis es el de crear un *framework* para el control de incubadoras neonatales que permita la incorporación de elementos como lo son sensores y actuadores que no fueron definidos a tiempo de diseño del equipo, lo que permitiría además un fácil reemplazo, la incorporación de nuevas funciones, prolongando así su período de vida útil. Como conclusión, con base en el trabajo realizado, es posible afirmar que la generación de un marco de trabajo que permita añadir y modificar los elementos que lo conforman, escalable y transferible, y al mismo tiempo cumplir con las normas para su implementación en centros de salud es posible.

La estructura del presente trabajo es la siguiente: El capítulo dos muestra el marco teórico incluyendo el estado actual de las incubadoras así como investigaciones recientes realizadas tanto para la incorporación de nuevas tecnologías en el equipo médico como el

acoplamiento débil y como este puede ser de ayuda para dar solución al problema planteado en la presente tesis. En el tercer capítulo se define la problemática, los objetivos y las metas. El cuarto capítulo describe el desarrollo de la metodología de tesis, detallando paso por paso el trabajo realizado. El capítulo quinto muestra la forma de utilización del equipo, los resultados y conclusiones del trabajo así como sugerencias para el trabajo futuro.

II. Marco teórico

2.1 Equipo médico

El equipo médico se refiere a aquel equipo destinado al diagnóstico y tratamiento de enfermedades, y rehabilitación de ciertas condiciones que afectan la salud de los humanos o cualquier otro ser vivo [1]. Generalmente su calibración, mantenimiento, reparación, capacitación a personal y diseño es atribuido a la ingeniería biomédica, para lo cual se hace uso de principios y técnicas de distintas áreas de ingeniería con un enfoque a la medicina, incluyendo conocimiento de áreas como lo son la eléctrica, mecánica y electrónica.

Cada área cumple una tarea importante, por ejemplo, la parte de eléctrica es la encargada del correcto suministro de energía, la parte mecánica de la construcción y diseño estructural del equipo para cumplir con requisitos de aislamiento y facilitar su posicionamiento por parte del personal. El área de la electrónica es la encargada de crear el control de los equipos, el manejo de señales y la creación de un sistema que permita el uso correcto de los instrumentos, así como ofrecer interfaces de calibración para el personal de mantenimiento. Tomando en cuenta tales aspectos que influyen y constituyen el equipo médico, se puede llegar a concluir que un dispositivo de este tipo puede ser considerado como un sistema mecatrónica, en el cual cada área o etapa cumple una parte importante del objetivo final.

La finalidad de las incubadoras neonatales es proveer las condiciones ideales para aumentar los índices de supervivencia de los recién nacidos, los bebés prematuros y sus necesidades; por lo tanto, estos son los primeros factores que deben tenerse en cuenta al diseñar este tipo de equipo. Los recién nacidos se encuentran en riesgo mayor de desarrollar hipoxia, hipotermia y muchas otras condiciones adversas debido a sus órganos no están completamente desarrollados para enfrentar una situación de supervivencia postnatal.

Uno de los problemas de mayor importancia es el de la termorregulación de los recién nacidos, debido a que estos deben de afrontar un cambio de temperatura en el ambiente desde el momento en que se encuentran fuera del vientre de la madre; si se presenta dicha situación por tiempo prolongado, en la cual el bebé pierde calor, es muy probable que esto derive en problemas de salud a largo plazo o incluso la muerte del bebé. Es por esto que la regulación térmica es una característica indispensable de este tipo de equipos médicos, además de otro tipo de funciones como el control de humedad con base en vaporización del agua almacenada en un depósito, o el de controlar el oxígeno a través de una válvula, dependiendo de las necesidades específicas de cada niño.

2.1.1 Incubadoras neonatales

Los bebés prematuros son aquellos nacidos antes de las treinta y siete semanas de gestación, por lo cual ciertos órganos no logran desarrollarse completamente, así mismo su sistema inmunológico puede ser muy débil, lo cual puede llevar a ciertas complicaciones e incluso llegar a la muerte si es que este se expone al mundo exterior de una manera no controlada. Es ahí donde las incubadoras neonatales (Figura 2.1) tienen un papel primordial.

La incubadora neonatal es un equipo cerrado, que consta de los siguientes componentes:

- Espacio para el recién nacido
- Capacete
- Colchón
- Base rodante
- Panel de control

Para cada parte del equipo existen varias características y normas, según el área que se encuentre, país y tipo de incubadora que se trate.

Por lo general, las características básicas con las que deberán contar son tener el suficiente tamaño para el individuo, el cual no lo limite y facilite el acceso al personal, un capacete transparente que permita el contacto visual con él y aislamiento del mundo exterior, incluyendo acceso para su posicionamiento. Además, deberá contar con un colchón que apoye el cuerpo de una manera pensada en la falta de soporte que pueda tener el paciente, una base estable y que facilite su transporte en caso de ser necesario (a menos de ser algún tipo de incubadora semi-estacionaria), y un panel que permita observar las condiciones (en caso de querer medirlas) en las que se encuentra.

Figura 2.1 Incubadora

La regulación de la temperatura es uno de los factores más importantes a controlar, debido a lo crítico que es para la estabilidad de los recién nacidos, esto debido al poco peso en el que pueden encontrarse, baja capacidad metabólica para la producción de calor o piel más delgada de lo normal [2]. Por lo general, el calor es brindado por medio del flujo de aire caliente, a través de un ventilador que toma aire del exterior y atraviesa un elemento calefactor, de una manera controlada para lograr la temperatura deseada. La medición de la temperatura puede estar basada en la temperatura ambiental dentro del capacete o en la piel del paciente.

La monitorización continua de la temperatura es fundamental para lograr el punto de equilibrio entre la temperatura actual y la deseada (de referencia). El intercambio de calor depende de varios factores, como los antes mencionados, la capacidad de aislamiento del equipo, y la humedad, entre otros.

2.1.2 Antecedentes

La historia de las incubadoras neonatales, así como de las técnicas para el cuidado de los recién nacidos, en especial de los bebés prematuros empezaron en el siglo XIX, siendo la primera incubadora introducida en 1835 por Von Ruehl en San Petersburgo, Rusia [3], la cual pretendía mantener la temperatura ideal para el desarrollo correcto de los recién nacidos. A partir de la fabricación formal de lo que es actualmente conocida como una incubadora en 1888 por W. C. Deming [4], se dió pie a un nuevo modo de ver y cuidar de los recién nacidos, estableciendo las condiciones que fomentaran su correcto desarrollo. Esto, disminuyó dramáticamente las tasas de mortalidad en los recién nacidos.

Para los años 1890s ya se podían encontrar incubadoras con una estructura muy parecida a la que se observa hoy en día a más de un siglo de distancia. El físico Alexandre Lion desarrolló una incubadora construida de metal, la cual contaba con un termostato y un sistema de ventilación el cual hacia uso de un simple abanico para mantener el flujo constante de aire. Sin embargo, para la época todo esto representaba un alto coste de producción, por lo que su implementación en hospitales públicos no se logró completamente.

Para 1934, Julius Hess, Jefe de Pediatría en el Hospital Michael Reese de Chicago [5], estableció las variables ambientales a investigar para los recién nacidos, añadiendo la variable oxígeno como otra parte importante a tomar en cuenta para el desarrollo ideal de los bebés, así como para ayudar como tratamiento para el síndrome de distrés respiratorio (SDRA), el cual se caracteriza por presentar síndromes inflamatorios, necrotizantes del alvéolo pulmonar y afectación de la circulación pulmonar [6], lo cual fue un primer paso en la introducción de ventilación mecánica o inducida a recién nacidos en situación de peligro.

La neonatología, parte de la pediatría que se ocupa del estudio y la asistencia de los recién nacidos, surgió, en parte debido al fallecimiento del hijo recién nacido del presidente de los Estados Unidos de América John F. Kennedy, en 1963 unos meses antes de su asesinato, debido a una complicación respiratoria. Esto permitió ver la necesidad de invertir en investigación y desarrollo de equipo de cuidado neonatal y dio inicio al avance en diferentes técnicas y tecnologías para llegar al punto en el cual se encuentra actualmente, en donde se cuenta con sistemas de control electrónicos y sensores no invasivos.

2.1.3 Elementos clave en el cuidado neonatal.

A pesar de que el objetivo de la incubadora neonatal siempre ha sido el mismo, se ha recorrido un largo camino para encontrar la mejor solución a los problemas presentados. A través del tiempo ha habido ciertos puntos y desarrollos clave, los cuales han cambiado las áreas de interés para los investigadores y científicos.

A continuación, se verán algunos de los más importantes y el avance que proveyeron en las distintas áreas de interés.

Control de la temperatura corporal

El mantener la temperatura idónea para el recién nacido fue una de las preocupaciones principales desde la invención de la incubadora. Pero fue hasta en los años 1950s del siglo pasado, que los investigadores se dieron cuenta de que uno de los factores por lo que los infantes no podían mantener su temperatura corporal fue el no tener suficiente grasa magra, por lo que además desarrollaban bajos niveles de glucosa en la sangre.

Alimentación

Así es como además de cuidar la temperatura, se empezó a cuidar la dieta de los infantes, proporcionándoles fórmulas o suplementos alimenticios especialmente desarrollados para ofrecerles una nutrición apropiada. Dichas fórmulas tenían una base de leche de vaca, a la cual se le agregaban proteínas, calcio, fósforo y sodio, las cuales fueron cambiando para las necesidades específicas de estos.

El problema que se presentaba, era que los infantes inmaduros tenían dificultades para consumir las fórmulas debido a lo inmaduro de su tracto gastrointestinal, por lo que Dudrick y Wilmore junto con el laboratorio de animales del Rhoads desarrollaron las bases para la infusión vía intravenosa de proteínas [7].

Visibilidad

Hasta la década de los 1940s la estructura de las incubadoras cubrían en gran parte al bebé, por lo que al introducir cúpulas de policarbonato, los doctores y enfermeras podían observar de una mejor manera el estado en el que se encontraba el infante, podían examinarlo de una manera más fácil y les permitía observar si se presentaba algún otro problema, sin comprometer la integridad del ambiente interior [8].

Soporte respiratorio

Debido a diferentes problemáticas y limitaciones tecnológicas hasta mediados de los 1960’s se empezaron a utilizar soportes de vida respiratorios. Anteriormente el medir el

oxígeno en la sangre era sumamente difícil y la única manera que se tenía de realizar esto era observar el color de la piel. Fue hasta dicha época en la que se logró medir el oxígeno a través de muestras de sangre. Además, se empezaron a utilizar máquinas de rayos X para mantener un régimen más controlado al momento de ayudar a los bebés prematuros con problemas de respiración, métodos que con el tiempo fueron reemplazados por otros más modernos y menos invasivos.

2.1.4 Tipos de incubadoras

Existe una gran variedad de equipos de incubadoras, cada una con un enfoque diferente. Entre ellas se encuentran:

- **Capacete cerrado.** Cuentan con filtros de aire minimizando el riesgo por infección y previenen la pérdida de humedad en el aire.

- **Doble pared.** Parecidas a las anteriores, pero con mayor aislamiento del exterior, en especial de la temperatura.

- **Servo-controladas.** Controlan automáticamente la temperatura y otras variables a partir de un valor de referencia dado y la medición de dichas variables a través de sensores.

- **Capacete abierto.** Proveen calor radiado desde la parte superior hacia la parte inferior y ofrecen un fácil acceso. Por lo mismo no son el tipo más aislado.

- **Portátiles.** Sirven de transporte cuando se requiere reposicionar al recién nacido a otra parte del hospital.

Por lo que dependerá de las necesidades del paciente el tipo de incubadora a utilizar.

2.1.5 Modelos comunes

En la región de Chihuahua se pueden encontrar diferentes marcas y modelos de incubadoras neonatales en los centros de salud, a continuación, se presentarán los equipos más comunes de encontrar y sus respectivas características, las cuales servirán para ilustrar los requerimientos y necesidades esperados para su utilización:

Saps ISOTERM

En sus dos variantes (cuidados generales y cuidados intensivos), pretende asemejar las condiciones intrauterinas, haciendo el control de temperatura ambiental y del infante, humedad relativa y concentración de oxígeno. En cuanto a la estructura, cuenta con una cúpula transparente y un sistema de circulación de aire que permite la entrada de oxígeno. Cuenta con alarmas audibles y visibles. El modelo de cuidados intensivos añade una pantalla para especificar el estado de las variables y un control de altura para el recién nacido [9].

Dräger Babytherm 8004/8010

Se presenta como una unidad de calor radiante, ya que, su objetivo es proporcionar calor a los neonatos prematuros y en término. Hace uso de calefactores cerámicos y de reflectores para proporcionar una sensación térmica adecuada y no unidireccional. Además, está diseñado para poder incluirse en otros equipos de la misma marca, como lo son sistemas arquitectónicos inteligentes, monitores de cabecera y sistemas de ventilación. A diferencia de la vista hasta ahora, esta es una incubadora de tipo abierto, esto quiere decir que no es necesaria una cúpula para la protección del bebe, en cambio cuenta con lámparas y reflectores en la parte superior direccionados al recién nacido [10].

GE Giraffe

Este equipo se puede definir como de gama alta y representa el actual estado del arte para cuidado de neonatos. Algunas de sus características son poder mantener la estabilidad de

la temperatura, aun si el panel es abierto, haciendo uso de impulsos de aire. Cuenta con un sistema que permite reposicionar al infante sin tener que tocarlo, ajustando automáticamente su cuerpo para reducir el estrés en él. Cuenta con la posibilidad de crear y conectarse a una red donde se pude realizar un registro del estado del bebé. Las variables pueden ser observadas a través de una pantalla [11].

Air-Shields Isolette C2000

Esta incubadora es capaz de controlar los parámetros de temperatura, oxígeno y humedad; ajustando el flujo de oxígeno dentro de la incubadora con una válvula y un sensor de oxígeno, y humedad a través de un sistema humificador, la cual es capaz de proporcionar una humedad relativa de 30 a 95 por ciento en incrementos del uno por ciento, La estructura cuenta con una altura variable [12].

FANEM Multisystem 2051

Al igual que la Dräger Babytherm, esta es una incubadora abierta, la cual cuenta con una lámpara fluorescente como unidad de calor radiante, un mezclador de aire/oxígeno y cuatro entradas: dos para O_2 y dos para aire comprimido, así como un sistema de ajuste de altura para la cama. También, cuenta con una lámpara halógena (propia de la marca) para realizar fototerapia, la cual sirve para disminuir la bilirrubina en el recién nacido [13].

2.2 Variables de medición

Las variables relevantes son cualquier factor, rasgo o condición que pueda existir en diferentes cantidades o tipos. A continuación, se describirá cada una de las variables ambientales de importancia en una incubadora.

La temperatura es una propiedad medible para el calor, representada en la escala de grados centígrados (°C) comúnmente en el sistema internacional. Es en la mayoría de los sistemas la variable más importante, esto debido a que provee las condiciones para las reacciones químicas, fermentación, secado y acondicionamiento de aire, por lo que tener un mal control de esta variable representaría problemas que lleguen a afectar a las demás

variables. Algunas de las razones por las que es difícil controlar la temperatura son la lentitud con la que cambia, la variación en el tiempo, multi zonas de control para las cuales se debe hacer un análisis de la dinámica del calor y la no linealidad que presenta.

El control de temperatura en el cuerpo es el proceso de mantener la temperatura constante (de 37°C), nuestro cuerpo solo puede mantener dicha temperatura si el calor que generamos es balanceado y equitativo al calor perdido.

La Figura 2.2 muestra la dispersión de calor en el cuerpo para el caso de que se encuentre en un área caliente (izquierda) y un área fría (derecha), reflejando en esta última como el cuerpo focaliza el calor en áreas con órganos vitales para ayudar a su supervivencia.

Figura 2.2 Dispersión de calor en cuerpo humano

Nuestro cuerpo controla la temperatura primeramente detectando el cambio externo de temperatura a través de receptores en la piel, al igual que los sensores en los sistemas de control, esta información es enviada al cerebro para tomar las decisiones adecuadas para mantener una temperatura ideal a través de los efectores (glándulas sudoríparas y músculos) al igual que los actuadores.

Una de las variables que afecta a la temperatura es la humedad, si la humedad relativa es muy poca, entonces el sudor será más efectivo para enfriar el cuerpo, en cambio, si la humedad es alta, el cuerpo requiere producir más sudor debido a que la evaporación es más lenta debido al vapor que ya existe en el ambiente.

La humedad relativa (RH), mide la relación de agua en el aire en forma de vapor y la capacidad que tiene el aire de sostener a dicha temperatura, y es de importancia para evitar

problemas respiratorios, alergias o infecciones. Este tipo de sensores requieren de un tiempo de establecimiento y de respuesta, al igual que el de oxígeno, pero en una cantidad menor de tiempo.

En cuanto al oxígeno, es de vital importancia la medición de la concentración en el entorno, la cantidad de oxígeno suministrado y la concentración de oxígeno en la sangre para saber con precisión las cantidades de oxígeno que requiera el paciente para no poner su salud en riesgo. Para la medición de esto último es necesario el uso de oxímetros con el parámetro SpO2, el cual es la medida del porcentaje de moléculas de hemoglobina unidas al oxígeno. Los niveles normales al medir el SpO2 son de 95 por ciento o más y un pulso alrededor de 70 a 80 latidos por minuto, en cambio los niveles que están entre el 94 y 80% son considerados como bajos y son signos de hipoxemia.

2.3 Normas y requerimientos

Para el desarrollo del controlador es importante conocer las normas y requerimientos propios del equipo para garantizar así su correcto funcionamiento, así como poder ser implementados en las distintas instituciones de salud. Es así, que la selección de componentes será basada conforme a la norma oficial mexicana NOM-066-SSA1-1993, la cual establece las especificaciones sanitarias de las incubadoras para los recién nacidos [14].

Dentro de las características mecánicas que indica la norma, establece que la cúpula o capacete (cubierta) debe tener las siguientes características:

- Debe ser de material que permita observar al recién nacido fuera de la cubierta, sin dificultad.

- Debe contar con lo necesario para facilitar el acceso al interior y colocar el recién nacido en forma adecuada. Si la cubierta puede abatirse o posee tapa debe estar diseñada de manera que evite su caída y apertura accidental.

- Los orificios para acceso deben de estar cubiertos cuando no estén en uso para que no se alteren las condiciones de operación de la incubadora.

- La cubierta debe tener 4 ventanillas, 2 al frente y 2 en la parte posterior para introducir los brazos. Debe contar con una tapa y otro sistema que permita sellar la ventanilla cuando no esté en uso.

Las características térmicas de la incubadora neonatal a tomar en consideración según la norma son las siguientes:

Tabla 2.1 Criterios de la Norma

5.4 Características térmicas a considerar en la incubadora (NOM-066-SSA1-1993)		
Punto	**Característica**	**Descripción**
5.4.1	Temperatura de equilibrio de la incubadora	El control de temperatura debe lograr este equilibrio desde 23 °C a los 37 °C.
5.4.2	Sobrepaso de la máxima temperatura controlada	Debe alcanzar un máximo de 39 °C, por medio de una acción especial del operador sobre el dispositivo de ajuste de temperatura. Este modo de operación debe ser indicado con una luz de advertencia u otra señal fácilmente reconocible o combinado con una indicación relevante para este sobrepaso de temperatura
5.4.3	Exactitud de la temperatura indicada °C	± 0.3 °C

5.4.4	Correlación entre la temperatura indicada y la temperatura de control(°C)	± 0.7 °C
5.4.5	Temperatura de las superficies que están en contacto con el recién nacido (°C)	40 °C máximo
5.4.6	Temperatura de las superficies accesibles al recién nacido	40 °C máximo
5.4.7	Temperatura en superficies metálicas	40 °C máximo
5.4.8	Temperatura en superficies no metálicas	43 °C máximo
5.4.9	Temperatura de las superficies accesibles al operador durante la operación normal de la incubadora (°C)	40 °C máximo
5.4.10	Temperatura en superficies metálicas de alta conductividad térmica	60.15 °C máximo

5.4.11	Temperatura en superficies de plástico de baja conductividad térmica o de madera	70 °C

2.4 ArquiTAM

El esquema de referencia de Taller Automático de Manufactura (ArquiTAM), está dirigido originalmente a coadyuvar en el desarrollo del sistema informático de control de pisos de producción. Propuesta en varios trabajos de investigación y desarrollada en varios trabajos de tesis [15], brinda una estructura que no solo resulta útil en pisos de producción, sino que es perfectamente escalable a sistemas con diferentes enfoques.

Para lograr lo propuesto se deberá crear un *framework* que permita la generación de modelos particulares a partir de un modelo general, siendo un *framework* otra manera más común a la cual referirse al modelo de referencia, el cual es un esquema ideado para el desarrollo y/o implementación genérica de una aplicación, obteniendo así una estructura como la mostrada en la Figura 2.3.

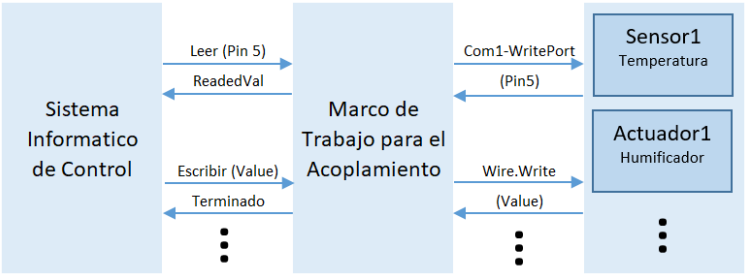


Figura 2.3 Marco de trabajo de acoplamiento y desacoplamiento con los elementos

En el diagrama anterior se representa el marco de trabajo a través de bloques y su intercomunicación con flechas y comandos. En los primeros dos bloques de izquierda a derecha se tienen las dos partes del sistema informático, el sistema base que sirve como lanzador de órdenes y se comunica con el esquema de referencia ArquiTAM con comandos genéricos como lo son **leer** y **escribir**. Luego este acoplamiento débil proporcionado por el anterior marco de trabajo permite comunicarse con los elementos sensor y actuador del último bloque a partir de funciones que cumplen con las particularidades de cada elemento como lo es el protocolo de comunicación utilizado para comunicar al sistema informático con sus elementos externos.

La aplicación del modelo de referencia a un caso particular de sistema de control se realiza bajo el concepto de *instanciación*, donde las clases se encuentran implementadas en el modelo de referencia y las instancias correspondientes a los elementos del sistema físico en particular son generadas a partir del modelo de particularidades (iMPR), todo ello con el soporte del paradigma orientado a objetos. Por tal motivo se establece una relación de instanciación entre modelo de referencia y modelo particular, relación equivalente a clase e instancia, elementos propios del POO.

En ArquiTAM, se plantean tres niveles para el desarrollo del sistema, tomando en cuenta los conceptos y detalles del dominio a nivel conceptual hasta la implementación de un sistema (Figura 2.4), correspondientes al nivel conceptual, implementación genérica e implementación particular respectivamente.

- **Arquitectura de referencia.** Requerimientos para el dominio y desarrollo de la solución.
- **Modelo de referencia**. Funciones y métodos para facilitar la instanciación del sistema para una aplicación particular.
- **Modelo particular.** Instancia creada a partir de las clases implementadas.

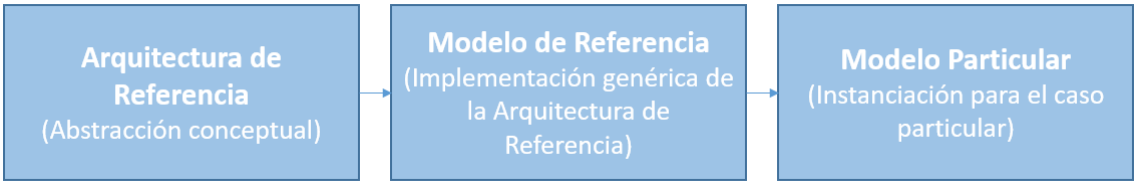


Figura 2.4 Etapas de abstracción ArquiTAM

A nivel arquitectura de referencia se plantea el uso de operación dirigida por modelo y acoplamiento débil como principios básicos para la flexibilidad del sistema de control.

Para la operación dirigida por modelo, se sugiere el uso de la técnica iMRP [16] para la abstracción de particularidades, en donde se propone manejar cada uno de los recursos del sistema en forma de nodo de manera que creen un grafo donde cada operación o proceso cuenta con los pasos necesarios para realizar el proceso, y de manera que cada elemento (ej. pieza, equipo y su operación) se encuentran en relación al *recurso tiempo* con el que cuentan. En este modelo se emplean únicamente dos tipos de elementos y se pueden observar mejor representados por el diagrama de la Figura 2.5:

- **Nodo** (recurso del sistema), el cual es abstraído en una clase base y posteriormente será dotada de atributos en el modelo de particularidades. Estos atributos para los nodos pueden ser de identificación (ID, Nombre, Nivel y Tipo).

- **Arco**, indican la relación que tienen los nodos. Estos dependen del tipo de grafo a los que corresponden (Recurso tiempo u Operación) en donde pueden incluir atributos de ubicación de los archivos.

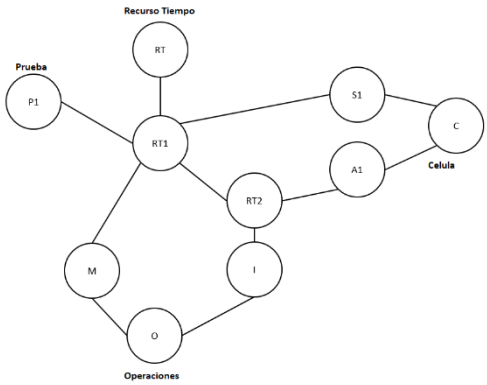


Figura 2.5 Modelo de particularidades

2.5 Acoplamiento débil

En 1967 James D. Thompson [17] presentó el concepto de acoplamiento débil basado en sus experiencias en el diseño organizacional donde él logró observar cómo organizaciones con entornos turbulentos y complejos promueven el desarrollo de estructuras más orgánicas y adaptables a los cambios. Creó un modelo basado en los trabajos existentes hasta la época, donde por ejemplo Talcott Parsons en 1960 [18] había desarrollado una estructura organizacional de tres niveles y le dió el enfoque de que todas las organizaciones deben ser sistemas racionales y naturales, abiertos y cerrados al mismo tiempo.

En [19] Orton y Weick promueven el uso del concepto de acoplamiento débil tanto a nivel técnico (cerrado, donde el acoplamiento fuerte produce estabilidad) como institucional (abierto a las fuerzas externas, donde el desacoplamiento entre elementos produce flexibilidad). Y definen términos de los cuales ArquiTAM posteriormente acoge para la creación de su modelo, dichos elementos siguen los lineamientos de la Figura 2.6 la cual alberga todos los elementos relacionados del acoplamiento débil considerados por su criterio después del análisis de diferentes puntos de vistas de distintos autores con interpretaciones que pueden diferir entre sí debido a las diferentes aceptaciones al ser un concepto particularmente metafórico.

Si bien la aportación de Thompson se enfoca en el área administrativa cuenta con la consideración de elementos análogos a los que se pueden encontrar en sistemas informáticos, tomando en cuenta las fuentes externas (generalización) e internas (interdependencia) y un análisis de la relación incierta que existe entre ambas en organizaciones complejas, y sin embargo, funcional a pesar de tan evidente disociación existente entre ellos.

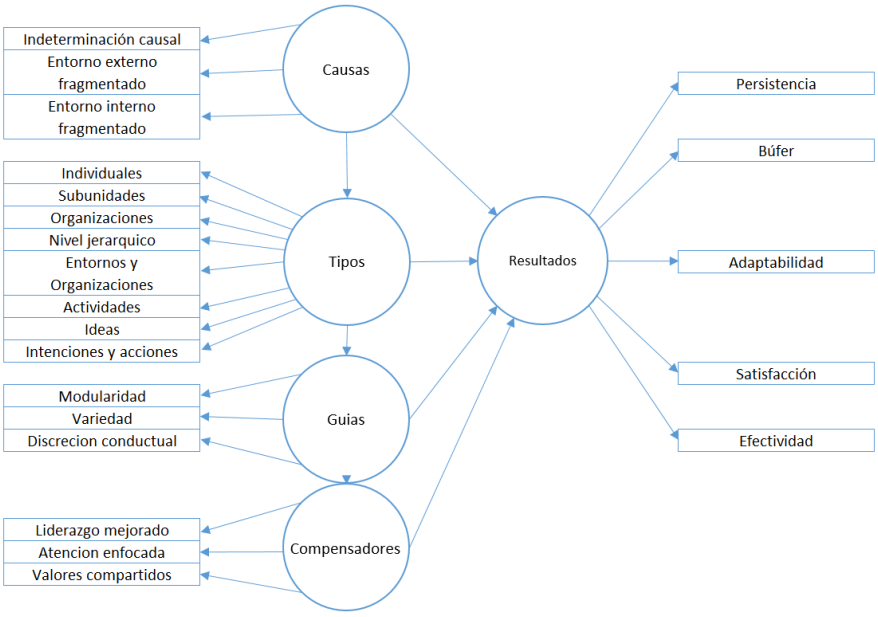


Figura 2.6 Teoría del acoplamiento débil [19]

El acoplamiento débil es aquella característica del sistema que facilita su reconfiguración ofreciendo la posibilidad de agregar/eliminar/modificar características no contempladas explícitamente en tiempo del diseño del sistema [20].

2.6 Programación orientada a objetos

La programación orientada a objetos es el paradigma de programación más utilizado en la actualidad [21]. Se puede referir a ella como un modelo que representa un subconjunto (abstracción) del mundo real, el cual cuenta con dos elementos: clases y objetos. Las clases son un prototipo de código el cual puede utilizarse para instanciar (reproducir) objetos iguales, se puede decir que es la estructura de un objeto y tienen atributos y métodos que definen a cada objeto en particular.

Si bien se podría seguir este paradigma en cualquier tipo de lenguaje de programación, existen algunos que facilitan su uso, entre ellos se encuentran:

- Objective-C
- C++
- C#
- Java
- Python
- Ruby
- Basic.NET
- ActionScript
- PHP

Tales lenguajes presentan características como el manejo de memoria automática, la abstracción de datos a través del lenguaje, estructura moldeada en objetos tomando como base sus estructuras de datos, concepto de clases, herencia y polimorfismo que pueden ser manipuladas a través del lenguaje.

2.7 Incorporación dinámica de código

En los sistemas operativos Windows existe la posibilidad de incorporar código dinámicamente, mediante el concepto de reflexión, haciendo uso de archivos DLL "Dynamic Link Library". El uso de archivos DLL ayuda a promover la modularización de código, reutilización de código, uso eficiente de memoria y reducción de espacio en disco, [22]. De esta manera gran parte de la funcionalidad del sistema operativo se proporciona mediante este tipo de bibliotecas de vínculos dinámicos. Para la utilización de este tipo de archivos existe una característica conocida como “Reflection” en Visual Basic e implementado utilizando el *namespace* “System.Reflection” la cual permite obtener objetos del tipo *Type* que describen:

- Ensamblados
- Módulos
- Miembros
- Parámetros
- Otras entidades manejadas en el código al examinar sus metadatos

Se puede utilizar para:

- Creación dinámica de una instancia de un tipo
- Enlazar el tipo a un objeto
- Obtener el tipo de objeto
- Invocar métodos, acceder a campos y propiedades del objeto

Como se puede intuir, esta función es clave para ciertas características del acoplamiento débil debido a que permite un nivel mayor de añadidura de código y al referirse a tiempo de ejecución sin modificar el controlador base es ideal para la implementación del mecanismo de acoplamiento débil propuesto en ArquiTAM.

2.8 UWP y Windows 10 IoT

UWP (acrónimo de Universal Windows Plataform) se refiere a la plataforma de Windows, la cual permite desarrollar aplicaciones para distintos dispositivos que utilicen el núcleo de dicho sistema operativo, de una manera flexible, y adaptativa. Es programable en C#, C++, Visual Basic y Javascript. Permitiendo desarrollar la UI (interfaz de usuario) en XAML, HTML o DirectX. De esta manera se pueden desarrollar aplicaciones fácilmente portables entre distintos dispositivos debido a la utilización de sistemas operativos que hacen uso de la misma base, teniendo así la posibilidad de crear software a nivel de usuario en todo el ecosistema creado por Microsoft (equipos de cómputo que funcionen sobre Windows, teléfonos inteligentes con Windows Phone, dispositivos inteligentes haciendo uso de Windows 10 IoT y consolas de videojuegos como Xbox One).

Windows 10 IoT es una versión reducida de Windows 10 orientada al internet de las cosas (concepto conocido en inglés como Internet of Things) enfocada a la creación de sistemas embebidos con estas capacidades. El ambiente cuenta con algunos requisitos como el uso de una PC con Windows 10 actualizado a una versión compatible con la versión de Windows IoT a utilizar para el desarrollo de sus aplicaciones, sin embargo, al seguir haciendo uso del núcleo de Windows permite desarrollar aplicaciones a través de herramientas como por ejemplo Visual Studio, tanto en C# como Visual Basic .Net, siendo el primero el más utilizado para su programación. Existen dos versiones del mismo sistema operativo, enfocadas a aplicaciones diferentes: Windows 10 IoT Core y Windows 10 IoT Enterprise, la primera está diseñada especialmente para el uso en CPUs ARM como es el caso de la tarjeta RaspberryPi y está enfocada al desarrollo de prototipos debido a la libertad y soporte que ofrece la otra

versión. Pero la principal diferencia o, al menos la más perceptible, se encuentra en la interfaz gráfica y la forma en que se ejecutan las aplicaciones, donde en el caso de la versión Core cuenta con una interfaz gráfica unificada para la plataforma universal de Windows, lo que hace que se cuente con una sola pantalla al mismo tiempo y no permite correr una segunda aplicación en paralelo. Por el contrario, la versión Enterprise permite el uso de WinForms y todas las demás funciones utilizadas en la interfaz gráfica de la versión completa de Windows, además de que permite correr varias aplicaciones al mismo tiempo (tomando en cuenta las limitantes de hardware del entorno en el que se trabaja).

También existe la versión Windows Server IoT, la cual es una versión completa de Windows Server y ofrece mayor manejabilidad y seguridad para las soluciones del internet de las cosas enfocada en aplicaciones industriales. La utilización de Windows 10 IoT para prototipos no tiene costo; en caso de querer comercializarlo cuenta con una cuota por dispositivo en caso de desear mantener el soporte de la plataforma.

La lista de placas con la que es compatible es la siguiente [23]:

- AAEON Up Squared
- DragonBoard 410c
- i-PAN M7/T7 CoverLens de Keith & Koep
- MinnowBoard Turbot
- NXP i.MX 6, 7 & 8M/8M Mini
- Raspberry Pi 2 & 3

Evidentemente el soporte actual es bastante limitado por el momento, pero hay que recordar que es una plataforma relativamente nueva y en constante desarrollo.

2.9 Tarjetas para el manejo de datos

Ya se ha hablado un poco acerca del sistema operativo a utilizar y el tipo de estructura que se emplea, por lo que es momento de dar una mayor descripción de las capacidades de las partes físicas que llevaran a cabo la etapa de procesamiento de datos y la implementación del sistema final.

CPU del Sistema informático

Como se mencionó en el punto anterior, se emplea la tarjeta Raspberry Pi, específicamente el modelo 3 B+ (Figura 2.7) haciendo uso del sistema Windows IoT y la plataforma universal de Windows. A continuación se muestran las características físicas de relevancia de dicha tarjeta:

- Procesador Cortex A53 de 64 bits
- Memoria RAM: 1 GB
- Puerto para memoria Micro SD
- Conexión inalámbrica: 802.11.b/g/n/ac wireless LAN y Bluetooth 4.2
- Puerto HDMI
- 4 Puertos USB 2.0
- Voltaje de alimentación: 5V
- Pines GPIO: 40

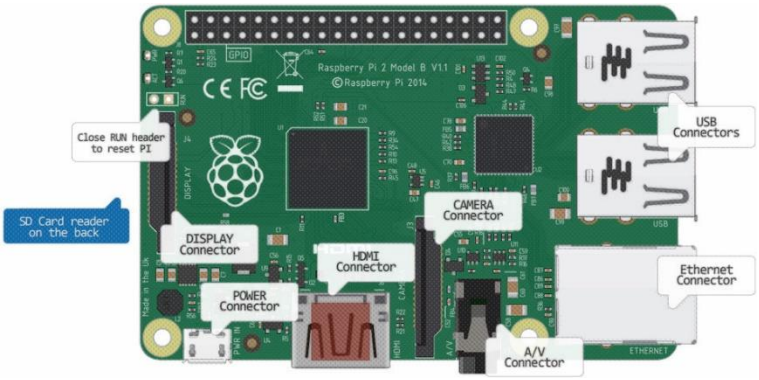


Figura 2.7 Tarjeta Raspberry Pi

Tarjeta para el manejo de entradas/salidas

Para el manejo de entradas y salidas se hace uso del Arduino UNO (Figura 2.8), el cual además de contar con convertidor digital/analógico con resolución suficiente para las lecturas, cuenta con pines independientes para la comunicación I^2C, además de contar con otros tipos

de protocolos de comunicación como lo son UART y SPI. A continuación se muestran las características físicas relevantes de dicha tarjeta:

- Micro controlador ATmega328P
- Pines I/O Digitales: 14 (6 PWM)
- Entradas analógicas 6 (12 bits)
- Velocidad de reloj 16 MHz
- Voltaje de alimentación: 5V

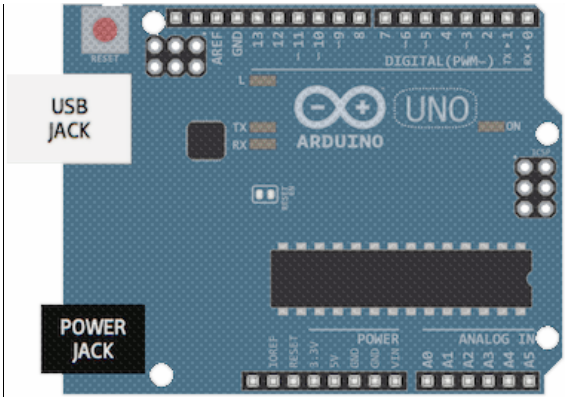


Figura 2.8 Arduino UNO

2.10 Revisión de literatura

A continuación se analizan y discuten trabajos y publicaciones científicas para ampliar el área de conocimiento acerca de los proyectos realizados en el campo de la medicina, específicamente para las incubadoras neonatales y de arquitecturas de referencia en sistemas informáticos.

2.10.1 Investigaciones enfocadas en el equipo médico

Existe una variedad de trabajos planteados en la dirección de incorporar nuevas funciones al control o agregar medición de nuevas variables. A continuación de muestran algunos de los reportes localizados en la literatura.

En [24] se presenta el diseño y construcción de un prototipo de incubadora controlado por lógica difusa en el Instituto Politécnico Nacional (2001).Dicha investigación se hizo basada en los requisitos de la Norma Oficial Mexicana NOM-066-SSA1-1993, la cual establece las especificaciones sanitarias de las incubadoras. A partir de esto, se realizó la construcción del chasis con su capacete y el colchón. Se creó un sistema de entrada de aire, así como de extracción, el cual consiste de orificios y ventiladores para la circulación de aire. Para la calefacción se utilizó una resistencia eléctrica donde se diseñó un sistema de humidificación, el cual hace fluir aire caliente para una dispersión más uniforme del calor. La medición de temperatura se realizó con cuatro sensores de temperatura, tres transductores LM35 para la medición del interior de la incubadora, y uno para la temperatura cutánea del neonato, y un microprocesador 6802 para el registro de las mediciones. En cuanto al sistema de control, este se desarrolló basado en lógica difusa, estableciendo un conjunto de reglas para representar el conocimiento del sistema, obteniendo la salida de control con base en el promedio de las señales de temperatura censadas.

Este proyecto logró mantener la temperatura como es establecida por la norma, así como la estructura, sin embargo, se menciona la falta de un sistema de inclinación de la plataforma.

En Diciembre del 2006 se realizó un prototipo de incubadora en la Universidad CES, de Medellín, Colombia. Dicho trabajo se presentó en la Revista de Ingeniería Biomédica en el año 2007 [25]. Este trabajo describe el proceso del acondicionamiento de la señal de alimentación, así como, la construcción del chasis y cúpula. En cuanto a sensores se incluyeron sensores de temperatura, humedad, luminosidad y ruido, los cuales indican si los parámetros se encuentran dentro del rango deseado. Además, cuenta con una alarma la cual se activa si la cúpula se mantiene abierta más de treinta segundos.

En este caso el proyecto parece haber sido más enfocado a la propia construcción de la estructura que al control de las variables, cumpliendo con las normas de seguridad, proporcionando el aislamiento al recién nacido con el medio ambiente común y proporcionando información acerca de las variables monitoreadas indicando si se encuentren o no fuera de los rangos establecidos, para ayudar al personal a tomar las decisiones convenientes.

En el 2015 en Quito, Ecuador se presentó una tesis que pretende funcionar como alternativa para el diseño del sistema de control de una incubadora. Al igual que el primer prototipo visto, este cuenta con un sistema de regulación de temperatura y un sistema humificador a partir del uso de una resistencia calefactora. Además, mide las variables de temperatura corporal, saturación de oxígeno en la sangre y ritmo cardíaco. También, se realizó una interfaz gráfica haciendo uso de una pantalla táctil para mostrar el estado de las variables. El autor de la tesis muestra de manera bastante completa los sensores y actuadores que incluye añadiendo los modelos de sensores utilizados y algunos modelos de incubadora que son compatibles con éstos, además, realiza las mediciones para su linealización y calibración.

Sin duda alguna un trabajo bastante completo y algo que lo diferencia de los demás trabajos es la medición de oxígeno en la sangre, en donde hace uso de un sensor especial para dicha función, el cual funciona observando los espectros fotométricos al pasar una solución por el extremo de la punta del sensor. También, se agregaron protecciones para sobre corriente y un respaldo de energía, algo con lo que cuentan los demás equipos comerciales, más no los demás prototipos vistos hasta ahora. Finalmente, agrega recomendaciones para su realización [26].

A través de diversos trabajos presentados, incluidos entre ellos los anteriormente descritos, se puede observar el interés por el desarrollo y mejora de equipos de incubadoras, así como de sus funciones, algunos enfocándose en el tipo de control manejado, otros en el aislamiento proporcionado y otros agregando funciones más complejas como la medición de oxígeno a través de sensores de nueva generación haciendo uso de procedimientos no invasivos. En lo que respecta a acoplamiento débil, los trabajos que se pueden encontrar son enfocados principalmente a manufactura y a procesos de automatización específicamente para líneas de producción o sistemas informáticos. Antes de poder analizar dichos trabajos se tiene que entender que es acoplamiento débil en el contexto dado por ArquiTAM, después se presenta una breve explicación y las conclusiones a las que se llegaron de algunos de estos trabajos.

Simons en 1994 presenta un marco de trabajo al cual llama “Levers of Control” (LOC) [27] y propone limitar el comportamiento de búsqueda así como delimitar el domino de actividad aceptable por parte de los participantes organizacionales en sistemas de manejo de control (MCS) los cuales se pueden definir como sistemas que transmiten información útil para ayudar a los gerentes en sus trabajos y en la toma de decisiones para lograr de manera eficiente y efectiva las metas organizacionales deseadas [28]. El introdujo cuatro constructores claves muy cercanos a los elementos del concepto de acoplamiento débil anteriormente visto, los cuales son: valores clave, riesgos a evitar, variables críticas e incertidumbres estratégicas, con los cuales en el trabajo presentado por Friederike [29] se muestra como existe una brecha de conocimiento y como el concepto de acoplamiento débil y marco de referencia puede variar su definición entre autores y los distintos niveles de control de los elementos que componen el

sistema. Para esto hace uso de un agente computacional que permite compensar las diferencias entre los elementos organizacionales y analiza los resultados de un sistema organizacional desde cero, mostrando las diferentes decisiones que puede tomar el agente para compensar las diferencias al momento de crecer y agregar nuevos elementos a la red.

En [30] se propone el mecanismo de acoplamiento débil para la integración de equipo de manufactura sin aplicar esfuerzo para su adaptación. Dicho trabajo hace referencia al modelo descrito por Orton y Weick el cual habla acerca del uso de compensadores para lograr el acoplamiento débil, como lo son el liderazgo compartido (delegando detalles de la implementación al equipo), atención de enfoque (enfocarse en lo esencial para la operación del equipo) y valores compartidos (uso de tecnologías de propósito general). Algo a resaltar de este trabajo es que para la parte del *driver* habla de manejo remoto de los dispositivos a través de servicios web, de esta manera son controlados desde un alto nivel a través de comandos genéricos.

En el modelo de referencia ArquiTAM, un trabajo más reciente de acoplamiento débil en la industria en donde se plantea la inclusión de diferentes sensores a través de un *framework* localizado en un sistema informático. Habla de cómo el mercado actual demanda tener una flexibilidad entre los elementos que componen al sistema. Igualmente hace referencia al internet de las cosas para el monitoreo remoto, a los sistemas reconfigurables de manufactura [31], los sistemas SCADA y a OPC (Ole for Process Control) para la comunicación de dispositivos sin la necesidad de modificar el sistema de control. Como se mencionó, hace uso de ArquiTAM y utiliza una metodología muy lineal siguiendo lo presentado en los trabajos de acoplamiento débil dados por J. Acosta y Sastron. Por la propia naturaleza del trabajo refiriéndose a la manufactura y control de piso hace uso de otra técnica llamada iMRP, la cual permite crear un modelo de particularidades, brindando así la posibilidad de crear una representación a través de grafos y nodos para la presentación de los elementos del sistema. Al final asegura que el modelo funciona correctamente para este tipo de problemas.

III. Planteamiento del problema

3.1 Descripción del problema

A medida que se trabaja con cualquier tipo de sistema, van apareciendo nuevas necesidades o errores que afectan a su funcionamiento. En el caso del equipo médico, los elementos que los conforman generalmente tienen un período de vida útil reducida, debido al trabajo constante al que se encuentran sometidos y los altos requerimientos que requieren para su utilización en el área médica. Por consiguiente, el mantenimiento suele ser constante y periódico, y muchas veces implica el reemplazo de componentes o insumos, los cuales solo pueden ser otorgados por la empresa con la cual se realizó la obtención de estos equipos y por nadie más, esta situación se vuelve más grave al considerar las limitaciones que ofrecen al momento de brindar la información de los modelos específicos de los componentes que integran al equipo, por lo que después de un tiempo pueden darse casos en los que sea prácticamente imposible conseguir un reemplazo de algún componente debido incluso también a la utilización de elementos propietarios otorgados por los fabricantes, por lo que en ocasiones recurrentes los centros de salud optan por el reemplazo completo del equipo.

El presupuesto para salud en el año 2018 es del 19 por ciento del presupuesto total destinado para “Desarrollo Social”, esto es un aproximado de 122,557 millones de pesos [32]. Esto hace que los recursos públicos disponibles para brindar servicios médicos a través de este tipo de instituciones sean limitados. En especial si tomamos en cuenta el precio de los diferentes equipos médicos que los hospitales necesitan para brindar el soporte básico a los pacientes. Al necesitar estar cambiando de equipo en períodos relativamente cortos es normal que se tienda por la adquisición de equipo de gama baja y por lo tanto con funciones limitadas, lo cual no permite tratar a futuro complicaciones más avanzadas a los pacientes y limita los servicios que se pueden ofrecer.

Tradicionalmente en el desarrollo de sistemas se maneja un enfoque de “acoplamiento fuerte” entre los módulos que lo componen, esto se refiere desde el punto de diseño con el que es concebido el sistema en donde se crea el sistema con base en elementos predefinidos los cuales no pueden ser reemplazados por otros diferentes. Creando así una estrecha relación entre el sistema y las partes que lo componen y dejando una inexistente flexibilidad para la actualización del propio sistema. Afortunadamente el panorama a futuro parece ir cambiando el tipo de enfoque a uno más “débil”, donde el sistema será hecho con base en estructuras genéricas actualizables [33], reduciendo así tiempos de diseño y trabajo al momento de generar plataformas a futuro, teniendo bases donde su enfoque no esté basado en el diseño de un sistema para una aplicación en particular, si no en el de uno genérico el cual permita incorporar con elementos de una manera modular.

Por lo tanto, el desarrollo de un controlador flexible, el cual permita el reemplazo de elementos del equipo por otros de iguales o diferentes característica, además de la inclusión funciones del equipo sin tener que reemplazarlo completamente o actualizando el sistema base del controlador, proporciona una gran ventaja en cuanto reutilización de hardware y software se refiere, vista no únicamente desde el punto del fabricante, el cual no se enfocará en la creación de un sistema dedicado si no en uno genérico que puede ser reutilizado a futuro en otras plataformas, sino también para los encargados del cambio de elementos facilitando su tarea y eliminando procesos lentos de actualización que en muchos casos del fabricante tendría que hacer directamente y por consiguiente nunca podrían ser realizados, reduciendo así costos y otorgando un alcance limitado no en origen si no en el trabajo futuro que se desee incorporar al marco de trabajo.

En el presente trabajo se describe la implementación de un sistema informático orientado al controlador de una incubadora neonatal, aplicando los conceptos de acoplamiento débil y el esquema de referencia ArquiTAM. A diferencia del enfoque tradicional del esquema de referencia antes mencionado, el cual se enfoca en pisos de producción, en esta ocasión se le da una orientación de incorporación de elementos sensores y actuadores gracias al sistema de tres capas visto en el punto 2.4. Se plantea su desarrollo utilizando el lenguaje de programación Visual Basic .NET para la plataforma universal de Windows IoT.

3.2 Objetivos

En este punto se expone la solución del problema presentado, demostrando la visión que se tiene con respecto al problema de investigación en cuestión.

3.2.1 Objetivo general

Diseñar e implementar un control de incubadora neonatal, que permita monitorear en tiempo real las condiciones en las que se encuentre el recién nacido que facilite la reconfiguración de los elementos de entrada y salida del sistema sin un esfuerzo de reprogramación del sistema.

3.2.2 Objetivos específicos

1. Identificar las Normas Oficiales Mexicanas (NOM's) que establecen las especificaciones, tanto sanitarias como estructurales, para el desarrollo de este tipo de equipos.

2. Definir una arquitectura de referencia dirigida a la flexibilidad del sistema de control.

3. Desarrollar un modelo de referencia con base en la arquitectura definida.

4. Diseñar un modelo que permita el monitoreo de manera local y/o remota de las distintas variables, así como su registro en una base de datos.

3.3 Metas de la tesis

1. Definir el dominio del conocimiento básico de incubadoras neonatales orientado al desarrollo del sistema de control.

2. Desarrollar un marco de trabajo para la generación de sistema particular de control de incubadora específica.

3. Diseñar una interfaz gráfica la cual permita verificar las condiciones en tiempo real.

3.4 Justificación y viabilidad

Una incubadora neonatal puede representar un costo aproximado de 3,000 a 50,000 USD dependiendo de los elementos con los que cuente, el nivel de cuidados requerido para el recién nacido y si esta cumple con los estándares, certificaciones de seguridad y construcción internacionales para operar en los centros de salud. Además, la funcionalidad de contar con acceso a Internet para compartir información acerca de las condiciones en las que se encuentra el recién nacido en tiempo real únicamente se encuentra en equipos de alta gama. Además de esto, hay que tomar en cuenta que los costos de mantenimiento pueden llegar a ser elevados y no resultar rentables a largo plazo.

Anteriormente, prototipos de incubadoras se han realizado como investigaciones de tesis al rededor del mundo, otorgando resultados satisfactorios aún con presupuestos ajustados. Estos equipos están compuestos de tres partes principales como lo son la estructura, el sistema de control y los sensores y/o actuadores. Siendo la parte estructural y el uso de algún tipo de actuador (en caso de ser necesario) los que requerirían de un gasto económico mayor, mientras que la implicación de crear una red de información a través de Internet y la parte de control digital no representarían un costo muy elevado por la parte de hardware implementando un sistema embebido para su manejo.

El presente desarrollo de un esquema de referencia como el planteado en el presente proyecto de tesis coadyuvará a implementar funciones de equipos de muy alta gama a precios asequibles para las instituciones públicas. Asimismo, los resultados obtenidos con la investigación y desarrollo de este proyecto podrán ayudar a dar inicio al diseño de equipo médico propio en la región.

IV. Desarrollo de la solución

4.1 Descripción de la solución

Como se mencionó en el capítulo anterior, se pretende crear un sistema informático base para el control de incubadora neonatal, que presente la flexibilidad para incorporar funciones o sensores no definidos explícitamente en tiempo de diseño, facilitando así la incorporación de funciones al controlador aun después de su implementación. Con este fin se utiliza el concepto de acoplamiento débil propuesto en ArquiTAM, que consiste en un mecanismo de tres capas descrito en el punto 2.4. En la Figura 4.1, se describe de manera sintetizada la estructura del sistema.

Entre las funciones que se podrían incluir, pero no se limita a ellas se encuentran:

- Monitoreo de temperatura ambiental
- Monitoreo de temperatura cutánea
- Monitoreo de oxigeno
- Monitoreo de humedad
- Control de temperatura

En la sección superior de la estructura del sistema, Figura 4.1, se tiene al controlador, el cual realiza funciones de coordinación con la rutina de enlistado de elementos instanciados, lectura de valores de sensores y la acción de control de salida para los actuadores cumpliendo así las funciones de lanzador de órdenes. En los siguientes niveles se encuentra implementado el mecanismo de acoplamiento débil. En el nivel inmediato inferior se tienen los objetos representantes de sensores/actuadores (clase cSensor y cActuador), en el nivel intermedio se encuentra el objeto envoltura (cSensorEnvoltura y cActuadorEnvoltura) y finalmente el propulsor (por ejemplo, el cI2C_Driver) que se encarga de manejar el detalle de los recursos físicos de la tarjeta para poder tener interacción con los elementos externos a ésta. La capa propulsora, como su nombre lo indica permite la conexión con otros dispositivos a través de diferentes protocolos como pudiera ser el protocolo I^2C; el usuario podría añadir otro módulo de comunicación permitiendo así utilizar otro protocolo de comunicación sin afectar a la estructura del sistema e incluso ser agregado a tiempo de ejecución del sistema (mediante incorporación dinámica de código) esto es posible en los tres niveles diferentes de la arquitectura.

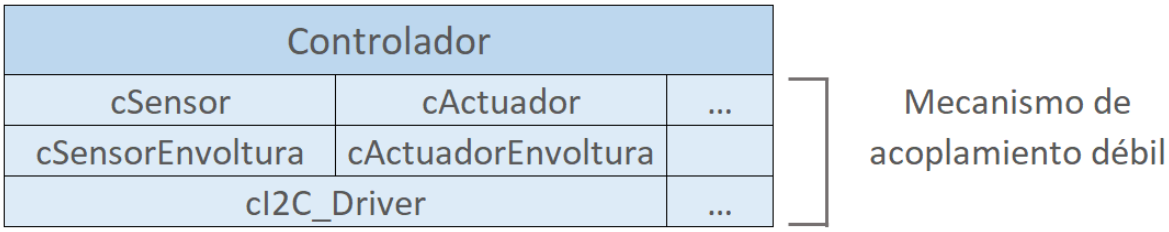


Figura 4.1 Capas del sistema informático

En un controlador débilmente acoplado con sensores/actuadores, introducir el manejo de más variables como pudiera ser el oxígeno y humedad se podrá realizar una vez terminado el modelo base y la comprobación de su funcionalidad. Tomando en cuenta que el acoplamiento débil permite añadir funciones que no fueron contempladas en tiempo de diseño, integrar nuevas funciones al sistema básico solo implica la caracterización de los elementos y la configuración en el sistema, sin requerir modificaciones mayores al código en el sistema base.

El análisis de la constitución propia del controlador se divide en dos partes, la primera parte se enfoca en el análisis de su estructura como sistema informático donde se crea el marco de trabajo e incorpora todas las partes inherentes a éste, como lo son el lanzador de órdenes, el esquema de arquitectura ArquiTAM y los módulos de trabajo, siendo la parte intangible y lógica del controlador. La segunda parte consiste en el análisis de la parte física de este, donde se observan las consideraciones de interconexión de los distintos elementos que conforman al sistema

4.2 Estructura del controlador.

El sistema se puede representar en una estructura a bloques en el cual cada bloque pertenece a un nivel diferente de aplicación. La figura 4.2 representa dichos niveles desde la parte superior donde se encuentra el usuario quien interactúa directamente con el sistema gracias a la interfaz gráfica del controlador hasta los sensores y actuadores que permiten observar y controlar las variables físicas que se encuentran en el medio ambiente. A continuación se describen las características de cada bloque:

Usuario. En este bloque se representa la persona que tiene interacción con el sistema, ya sea a nivel de personal del hospital, donde su interés es el de observar el estado de las distintas variables que maneja el sistema, o a nivel de técnico el cual es el encargado de configurar nuevos o existentes elementos del sistema, a través de un sistema de creación y configuración de elementos, los cuales pueden ser entradas y/o salidas del sistema esto es, elementos sensores o actuadores respectivamente. Además de poder tener acceso a una pantalla de configuración del controlador, donde pueden modificar parámetros como el tipo de acción de control a utilizar. En este nivel de manejo del equipo se debe contar con cierto conocimiento de los elementos del sistema, como el tipo de señal a leer, la comunicación a utilizar o la ubicación del mismo en el sistema y elementos que no son de uso común para el personal médico del centro de salud. Es así que teniendo estos dos tipos de usuarios (personal del hospital y personal técnico) como se puede crear una sinergia entre las necesidades que se presenten en el centro de salud y la solución que se le puede otorgar.

RaspberryPi. Es la parte central del sistema, cuenta con la lógica para el manejo de recursos del sistema. En un nivel superior se encuentra el usuario, para el cual presenta una interfaz gráfica fácil de leer y manejar, dándole acceso tanto a los valores leídos por el equipo, como a los objetos que lo integran desde una parte lógica. En este bloque se encuentra implementada la base del sistema informático desarrollado en un ambiente de programación orientada a objetos, con la rutina de control y las rutinas de lectura y escritura. La integración de estas rutinas con los sensores/actuadores se realiza bajo el mecanismo de acoplamiento débil, de la manera propuesta en el esquema ArquiTAM, Figura 4.1. El mecanismo cuenta con tres niveles diferentes que permiten integrar de manera flexible el sistema base con los elementos particulares de éste (sensores y actuadores). Es importante resaltar que en el diseño del sistema base no se especifican explícitamente características de sensores/actuadores (cantidad, tipo, protocolo, entre otras), por lo que será tarea del técnico-usuario la integración de nuevos sensores/actuadores, con base en la descripción de sus características a partir de una nueva instancia. Las tres capas que permiten dicho desacoplamiento del sistema son: capa representante, envoltura y propulsor. La clase **representante** permite tener una estructura genérica de los parámetros necesarios para manejar elementos sensores/actuadores, luego la **envoltura** permite tener un lenguaje común o comandos homogéneos para su operación. Así, la estructura base comanda a los elementos del sistema de manera independiente de las características específicas del elemento, lo cual permite manejar los recursos físicos integrados al sistema base (Tarjeta Raspberry). A través de este mecanismo (objetos en tres capas) el sistema informático de control (Base) interactúa con el nivel de bloques que maneja a los recursos físicos, entiéndase como recursos físicos a los protocolos de comunicación con los que cuenta la tarjeta y sirven para establecer una comunicación ya sea con sensores/actuadores conectados directamente a la tarjeta Raspberry o bien con otro tipos de placas auxiliares para el manejo de los mismos (por ejemplo: Arduino).

Arduino/ADC. Con la intención de mejorar la capacidad del sistema con respecto a la capacidad obtenida con el uso únicamente de la tarjeta Raspberry, en el presente sistema se propone una tarjeta Arduino, que funciona como una tarjeta de expansión de puertos, tanto de entradas como de salidas, ya que su interfaz es mucho más amplia que la manejada por la RaspberryPi, especialmente hablando de entradas analógicas y salidas PWM, además proporciona una ventaja extra con respecto al manejo directo de las entradas y salidas del sistema en la tarjeta RaspberryPi, la cual es otra etapa de aislamiento (protección) de sistema para que en caso de que exista algún problema con la tarjeta, esta no afecte a todo el sistema. Esta situación no presenta una limitación al sistema, ya que puede utilizarse cualquier tipo de ADC o tarjeta de adquisición de datos para este propósito. Generalmente las tarjetas ADC hacen uso del protocolo I^2C para la comunicación con las tarjetas encargadas de la lógica, lo que al poder hacer uso de otras tarjetas como lo es Arduino aporta una ventaja añadida de poder hacer uso de otro tipo de protocolos de comunicación, siempre y cuando las dos tarjetas cuenten con los medios físicos para llevar a cabo la comunicación entre ambas y se desarrolle el módulo de comunicación a nivel propulsor en la RaspberryPi.

Sensores/Actuadores. En este bloque se representan todos aquellos elementos que permitan tener una interacción con el medio físico, en el caso de los sensores midiendo las magnitudes de las distintas variables que lo rodean y en el caso de los actuadores controlando dichas variables, entendiendo como variables a todos aquellos parámetros físicos del entorno, incluidas entre estas temperatura, humedad, posición, presencia, velocidad o potencia. Estos pueden generar una señal de diferente tipo, por lo cual es necesario el uso de puertos como los presentes en el Arduino/ADC tanto para lectura o escritura, dichos valores al ser leídos son enviados a la tarjeta RaspberryPi para su manejo o al contrario de la tarjeta antes mencionada hacia el exterior como una señal de control.

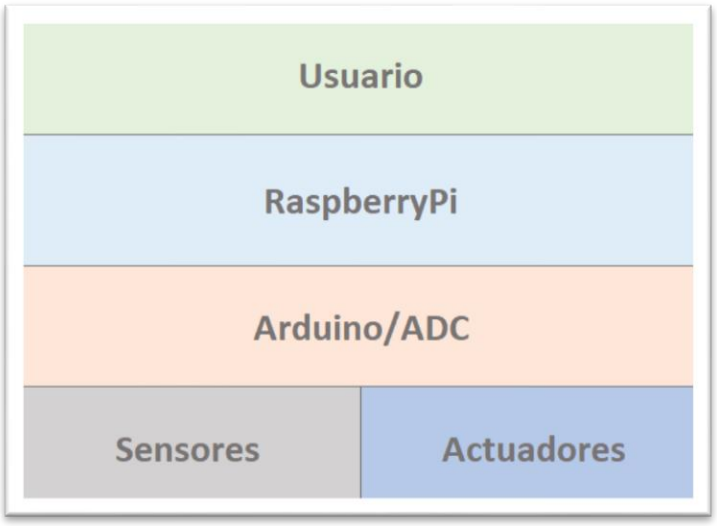


Figura 4.2 Estructura del sistema

Como se ha mencionado en este mismo punto, la tarjeta Raspberry es la parte central del sistema, en donde se encuentra el hardware del sistema interactuando con los demás elementos del sistema tanto directa como indirectamente en el caso del último nivel del esquema a bloques de la Figura 4.2. En la Figura 4.3 se muestra una representación de la lógica/proceso del sistema.

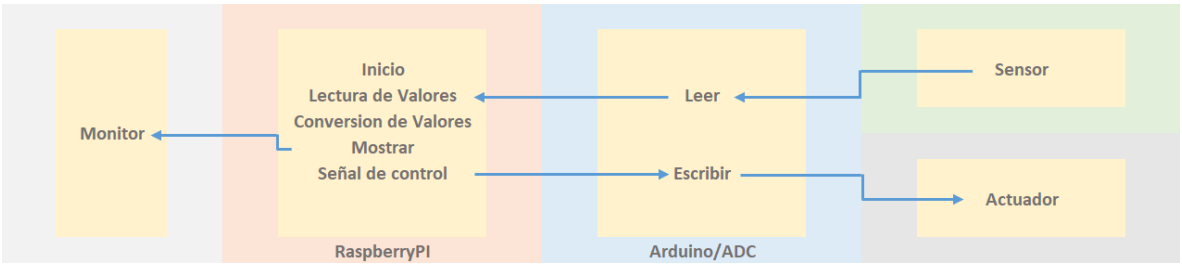


Figura 4.3 Abstracción del funcionamiento del sistema

El monitor o pantalla en el inicio de la operación no cuenta con ningún parámetro leído, solo teniendo una pantalla de bienvenida para el usuario y el menú de configuraciones. El sistema inicia con una colección modificable de los elementos del mismo, se cuenta con una lista de sensores y actuadores ya configurados. Se recorre la lista de objetos guardados en la colección, por lo que para cada sensor la Raspberry solicitará al Arduino la medición del sensor (lectura). El Arduino obtiene el valor del sensor y lo envía a la Raspberry, una vez obtenido el valor medido por el primer sensor se convierte por medio de la ecuación de segundo orden formada gracias a los parámetros otorgados por el usuario al momento de la configuración del sensor en unidades de temperatura, humedad, etc… (ver punto 4.7.4 para más detalles acerca de la conversión de unidades).

Una vez obtenida la variable leída y convertida a las unidades deseadas, se compara con el valor de referencia introducido previamente por el usuario (ejemplo: se tiene que el valor leído de temperatura es 19°C, mientras que el valor deseado o Set Point es 25°C, por lo tanto al existir una diferencia entre la temperatura medida y la deseada se aplica una acción de control según el usuario haya definido al momento de la configuración del controlador). El valor de la diferencia (error calculado) es utilizado por la acción de control seleccionada para definir la señal de control a aplicar al actuador respectivo, a través de la placa Arduino y la etapa de potencia correspondiente. El valor leído y convertido así como el valor de la variable a controlar es mostrado en la pantalla o monitor de la incubadora. Este procedimiento se realiza para cada elemento que contiene la lista de sensores (variables).

La interfaz de usuario es adaptable a modificaciones en el conjunto de elementos respecto a su despliegue de manera gráfica y en una sola pantalla, con la finalidad de mostrarlos de manera fácil e intuitiva para su identificación y análisis. Es importante destacar que la actualización de valores en pantalla, así como la resolución de los valores medidos son conformantes con las normas mexicanas para su utilización en centros de salud.

4.3. Estructura física del controlador

En la presente sección se describirán los componentes físicos que integran del sistema, Figura 4.4.

Como se menciona en el punto 4.2, los principales módulos del sistema son:

- **Raspberry Pi**: Tarjeta encargada llevar a cabo el proceso del sistema, lanzador de órdenes, ArquiTAM, conversión de valores leídos e impresión de los mismos en la pantalla, así como la configuración e instanciación de los elementos sensor, actuador y controlador.

- **Arduino/ADC:** Tarjetas con entradas y salidas analógicas para la lectura y escritura de datos (lectura de sensores y señal de control a actuadores). Puede ser reemplazado por otro tipo de tarjeta según las necesidades sin tener que cambiar el código del sistema.

- **Sensores/Actuadores:** Elementos que pueden ser cambiados por otros de iguales o diferentes características para la lectura y control de variables dentro de la incubadora.

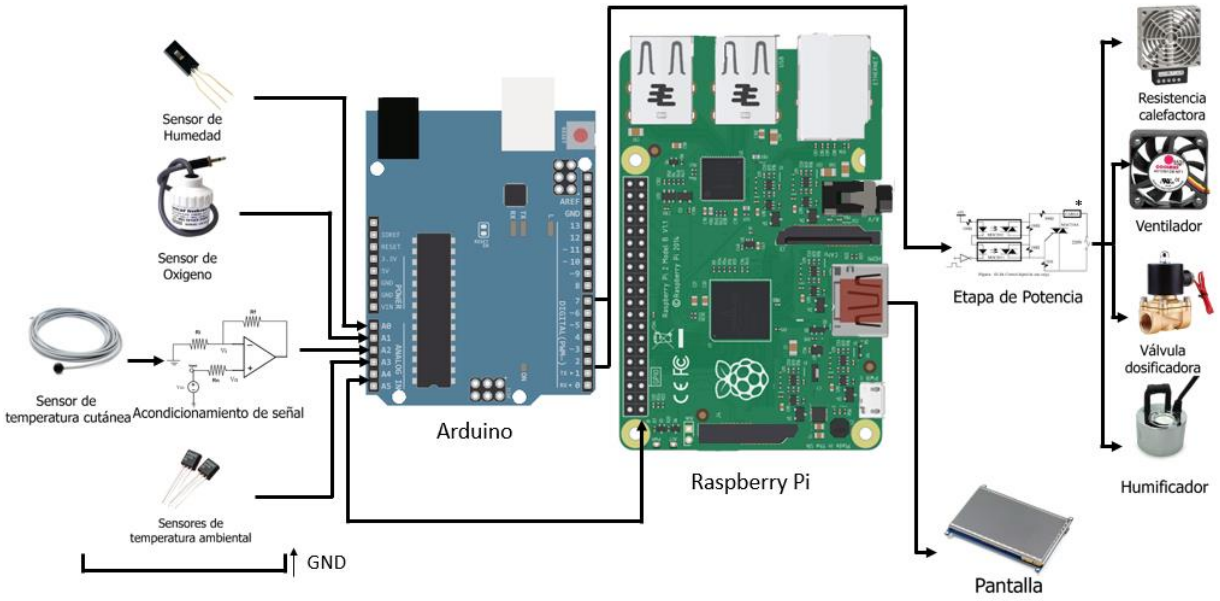


Figura 4.4 Diagrama Esquemático

En la Figura 4.4 se muestra el diagrama esquemático del sistema con los distintos elementos, así como la interconexión entre los mismos. En el esquemático se incluyen los distintos sensores que fueron utilizados para pruebas reales del equipo, algunos de estos requieren de una etapa de acondicionamiento previa al ingreso de su lectura por el Arduino.

Posterior a la etapa de acondicionamiento de la señal la señal es conectada a la entrada de la tarjeta Arduino (previamente configurada). Las señales analógicas son convertidas a señales digitales las cuales son enviados a través del protocolo de comunicación correspondiente, en el presente caso la comunicación entre la tarjeta Raspberry y el Arduino es realizada pero no limitada a I^2C, luego dichos valores son manejados según las particularidades especificadas por el usuario en la Raspberry. Los valores medidos pueden ser desplegados en la pantalla a través del puerto HDMI de la Raspberry. Así mismo, los valores medidos son utilizados por los algoritmos de control respectivos para generar las salidas de control comunicadas, a la etapa de potencia para finalmente hacerla llegar al actuador. A continuación, se presenta una descripción a mayor detalle a los elementos que conforman al sistema.

4.4. Integración (acoplamiento) del Sistema.

En la figura 4.5 se muestra la estructura del sistema destacando las tres capas que implementan el mecanismo de acoplamiento débil propuesto en el esquema ArquiTAM, representante, envoltura y propulsor, para la integración de sensores/actuadores.

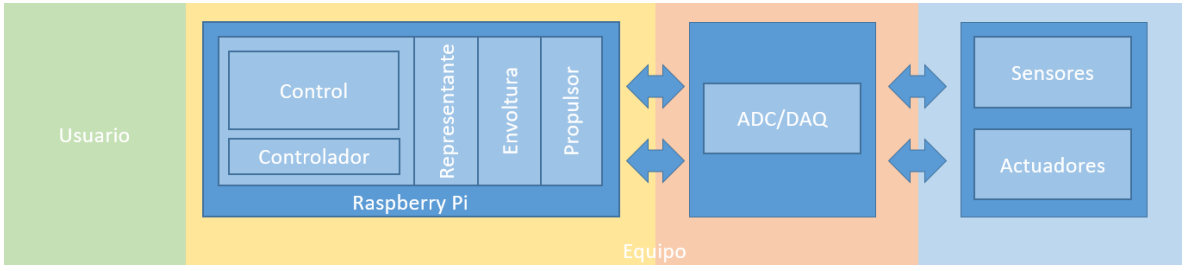


Figura 4.5 Estructura del sistema con sus distintos niveles

El acoplamiento de los distintos módulos o secciones del sistema se describe haciendo la abstracción del mismo en dos módulos o secciones: a) Supervisión/Control y b) Sensores/Actuadores.

a) **Supervisión y control.** Cuenta con un ciclo de trabajo que se encarga de realizar todas las operaciones necesarias para el funcionamiento del sistema, entre ellas leer sensores, convertir y actualizar valores leídos, realizar la acción de control correspondiente, enviar la señal de control y actualizar valores en la pantalla de lectura de datos. Se encuentra implementado a manera de *framework* en la clase control encargado de lanzar órdenes que permitan la utilización de sensores y actuadores, en forma genérica, sin contar con ninguna característica o parámetro de estos, haciendo uso de comandos que permitan realizar las acciones básicas necesarias. Una de estas acciones es la de control implementado en la clase controlador.

   Gracias a la estructura genérica o *framework* se logra la comunicación con el Arduino el cual es el medio físico de lectura de los sensores, dicho valor como es mencionado en el punto 4.2 es convertido a las unidades deseadas por el usuario, simplemente sustituyendo dicho valor en la ecuación de segundo orden obtenida a través de los parámetros para formar el polinomio que define el comportamiento del sensor. El valor obtenido, por ejemplo en grados centígrados, es comparado al valor deseado por el usuario al momento de configurar el elemento controlador. La parte del controlador define la acción de control a realizar, y se implementa en una clase separada del control. Esta utiliza al valor del sensor leído, la acción de control a realizar y la variable de control. En resumen se puede decir que lee la referencia o Set Point, obtiene el valor real de la variable a controlar, calcula el error, y se aplica la acción del controlador para definir el valor de la variable de control a aplicar al actuador. La acción de control es decidida según el módulo de control seleccionado por el usuario, ligando al sensor del cual se medirá la variable con el controlador al cual se le enviará la señal de control.

   El controlador define en la acción de control a implementar. Por ejemplo, en el caso de un controlador proporcional se multiplica la diferencia obtenida entre la temperatura medida con la deseada por una ganancia establecida por el usuario, y finalmente dicho valor obtenido del controlador el enviado al Arduino para posteriormente ser enviado al actuador pasando por su respectiva etapa de potencia.

   El proceso de control de la variable se realiza de la siguiente manera. El Arduino realiza la medición de la variable, por ejemplo, con un valor de 550, este valor es enviado a la Raspberry Pi y es introducido al polinomio para su conversión, de acuerdo con la información proporcionada por el usuario sobre las características del sensor, esto es para obtener la ecuación como la siguiente: y=x/30+5, al sustituir el valor de 550 se obtendría el valor en las unidades configuradas, en este caso, 23.333. Este valor es enviado a la clase controlador, la cual es configurada según las especificaciones del usuario, todo ello se repite cada 200 ms. En caso de que se haya seleccionado (y se cuente con el módulo de controlador) un control proporcional, el valor convertido es

comparado con el valor de referencia dado por el controlador, dependiendo del error se le aplica una ganancia para así de esta manera obtener una señal de control la cual es enviada al actuador a través de a una etapa de potencia y así actuar sobre la variable de interés.

b) Sensores/Actuadores. Los bloques sensores y actuadores, ubicados en la última sección de la Figura 4.5 representan diferentes tipos de mediciones y/o variables, como pueden ser:

- Sensores
    - Temperatura cutánea
    - Temperatura ambiental
    - Oxigeno
    - Humedad
    - …
- Actuadores
    - Resistencia Calorífica y Disipador
    - …

Donde los puntos suspensivos indican la flexibilidad del sistema para agregar otros elementos a dichas secciones.

De acuerdo a la propuesta para acoplamiento débil planteada en ArquiTAM, la integración de sensores/actuadores se realiza mediante tres capas cada una implementada en una clase: Representante, Envoltura y Propulsor.

- cSensor

Esta clase corresponde al elemento representante, contiene una abstracción de un elemento sensor en forma genérica con las propiedades: ID, Nombre, Unidades, Dirección, Elementos del polinomio característico. Además, contiene los parámetros para la rutina de inicialización de comunicación donde se encuentran elementos como la dirección del esclavo

(en caso de utilizar I^2C), un identificador y la configuración de velocidad del puerto. Finalmente, se encuentran los prototipos de las funciones que son utilizadas para el manejo la medición como Leer o Escribir.

- cSensorWrapper

Esta la clase corresponde a la capa envoltura y permite la comunicación entre la capa representante y la capa propulsor. Facilita la adaptación a las funciones propias del propulsor (driver), utilizando características particulares del mismo como nombre de parámetros y funciones propias del mismo, necesarias para la comunicación con el sensor. Atiende las particularidades requeridas para implementar los comandos Leer y Escribir manejados a nivel objeto representante.

- cI2C_Module

Maneja los recursos del medio físico de comunicación con el sensor. Esta clase implementa las funciones requeridas por el medio físico de comunicación para la comunicación con el sensor, ya sea un protocolo de red (en este caso I2C) o bien en forma directa conectado a la tarjeta (Arduino).

En el caso de acoplamiento de los actuadores, las tres capas de acoplamiento son las siguientes: cActuador, cActuadorWrapper, cI2C_Module. Su función es equivalente a la función de las tres capas en el acoplamiento de sensores. En el caso del objeto representante del actuador, la clase actuador con los parámetros: ID, PIN (o dirección a donde se enviará la señal de control). La clase envoltura, donde se crea el enlace para la utilización del comando genérico Write para la escritura en cualquier tipo de actuador y la configuración de manera genérica para el protocolo de comunicación (comunicación con el propulsor). Finalmente se encuentra la capa del propulsor, en este caso, debido a que se utiliza el mismo canal de comunicación (I2C), se utiliza el mismo recurso que para el sensor (cI2C_Module), el cual puede ser reemplazado por el del protocolo deseado creando un módulo para dicho protocolo.

La figura 4.6 muestra los distintos parámetros que se solicitan para la configuración de los sensores y actuadores. En el caso de los sensores se pide definir un ID, Dirección, Nombre, las unidades de medición, tipo de conexión, y las constantes que conforman a la ecuación de segundo orden (a través de un polinomio), y un ID y Pin para el Actuador.

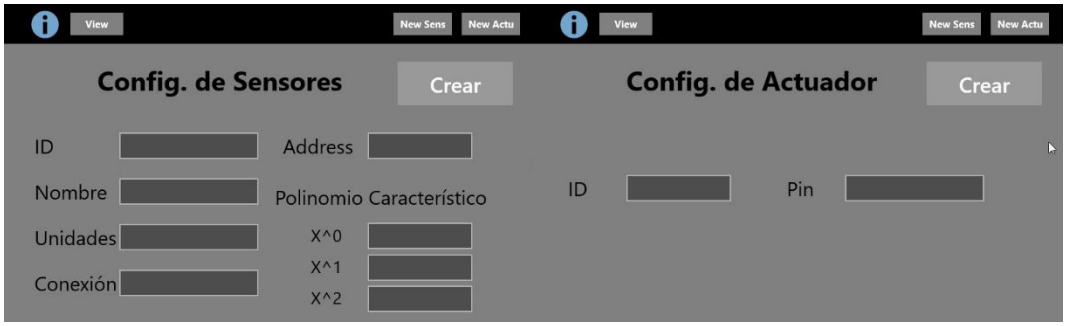


Figura 4.6 Pantalla de configuración de sensores/actuadores

En la figura 4.7 se muestra una abstracción de la estructura del sistema haciendo uso de constructores del estándar UML, [34] y [35].

Entre la clase control y C_In/C_Out se encuentra el modelo de tres capas definido por ArquiTAM, Representante, Envoltura y Propulsor respectivamente para los sensores y actuadores. La única diferencia entre estos son los parámetros que manejan, pero con la misma estructura.

La flexibilidad obtenida por ArquiTAM al agregar un nuevo sensor empieza por la etapa de la clase representante, la cual define que ordenes serán empleadas para el manejo del objeto sensor y establece cual es la siguiente etapa de abstracción. Al crear un objeto sensor se establecen las propiedades inherentes al mismo y requeridas para su manejo (Identificador, Dirección física, Parámetros para la ecuación de conversión de los valores leídos a valores en unidades estándar (°C, % Humedad, Kg, etc…), Tipo de comunicación a emplear), estos parámetros son manejados en la capa de envoltura, la cual sirve como interfaz entre el sensor particular y el sistema genérico. La última capa (propulsor) del modelo actúa como puente entre la interfaz física (para la lectura y escritura de valores) y la interfaz de software para poder manejar con los comandos y parámetros de comunicación establecidos al sensor. Este último puede ser considerado como un módulo de comunicación, el cual puede ser reemplazado por otro para incorporar alguno tipo de comunicación diferente.

Lo mismo sucede con los actuadores con la diferencia de los parámetros y comandos que maneja, por ejemplo, este no requiere de una ecuación de conversión ya que no lee valores, y en vez de utilizar un comando de lectura como lo puede ser “Read” para leer el valor del sensor actual, se utiliza un de escritura como “Write” para tener una salida de control hacia los actuadores.

La Figura 4.3 ilustra el proceso de manera general, donde al crear un sensor / actuador en la clase Control, dicho objeto se almacena en una colección dependiendo del tipo de objeto creado donde se almacena él y todas sus particularidades, físicamente hay que realizar la conexión del sensor / actuador con el Arduino donde para cada entrada /salida corresponde una dirección (Ejemplo: para la entrada analógica 0 la dirección es 0xE0) la cual es la dada al momento de configurar el objeto por el usuario. La clase controlador por su parte liga las lecturas de un sensor con la señal de control de un actuador seleccionado por el usuario, y define la acción de control a emplear, luego se continúa el proceso por el lanzador de órdenes (lectura de sensores, conversión, escritura de actuadores, impresión de valores en pantalla de usuario). De esta manera las clases Controlador (cOn/cOff, cPID...) igualmente funcionan como módulos para agregar nuevos métodos de control según se desee.

En resumen, para agregar un nuevo sensor o actuador al sistema simplemente se configura a través del llenado de sus características en su respectiva pantalla, y en el caso de querer agregar nuevas capacidades de comunicación y de acciones de control el usuario debe incorporar los módulos de comunicación y controlador respectivamente ya sea a través de incorporación dinámica de código a través de DLL’s utilizando Reflection sin tener la necesidad de interrumpir al equipo ni cambiar el código del sistema.

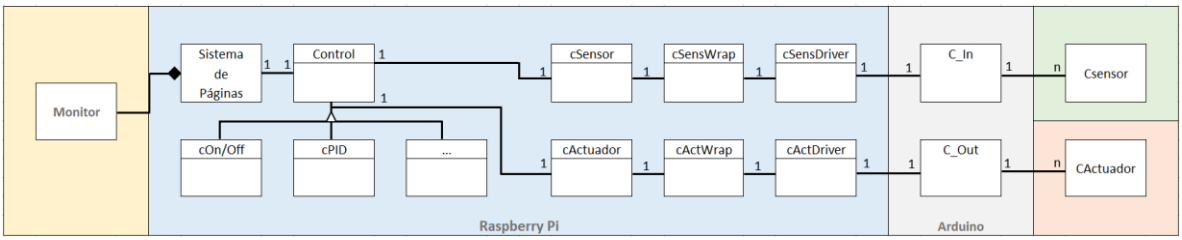


Figura 4.7 Diagrama de clases

4.5 Sensores

Los sensores son aquellos elementos que permiten medir o recibir señales del entorno, y por lo tanto existen en una gran cantidad y con diferentes características. La estructura presentada en la Figura 4.4 muestra algunos ejemplos de sensores que nos permitirán realizar las pruebas del equipo, pero esto no quiere decir que el sistema se encuentre limitado al uso de estos, si no, muestran una pequeña parte de las posibilidades que se tiene al contar con una estructura de este tipo, donde no se tenga ningún parámetro especifico definido de los elementos que conforman al sistema, a continuación se describen los elementos que integran la estructura de la figura anteriormente mencionada (Figura 4.7).

Medición de temperatura ambiental (LM35) [36]

Se propone el uso de sensores LM35 para la medición de temperatura ambiental debido a su bajo coste y alta precisión, en conjunto con un sensor de temperatura de piel de uso biomédico, el cual estará en contacto directo con la piel del recién nacido, sirviendo el primero como un valor de referencia para el control de ambas temperaturas, en especial la de mayor interés, la temperatura del bebé.

El sensor de temperatura a utilizar, LM35 (Figura 4.8), tiene tres pines, de los cuales dos son de tierra y Vcc, y el tercero para el valor de temperatura en milivolts, dichas conexiones pueden ir conectadas directamente a los pines correspondientes de la tarjeta. Al tener un tiempo de respuesta prácticamente instantáneo se puede medir en cualquier momento. Algunas de sus características son:

- Calibrado en Celsius
- Lineal
- ± 0.5 Precisión a 25°C
- Rango -55°C a 150°C

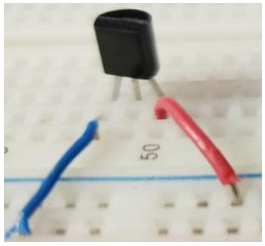


Figura 4.8 Sensor de temperatura ambiental

Temperatura de piel (Termistor YSI 45000) [37]

Este es un termistor (Figura 4.9) de 10 KΩ a 25°C, por lo que para medir su valor resistivo será necesario hacer una configuración muy parecida que la del LM35 (con la diferencia de solo contar con dos terminales), agregándole una resistencia de 10KΩ en serie entre este el extremo de una de sus terminales y tierra, y tomando la medición en el nodo creado entre estos. Algunas de sus características son:

- Calibrado en Celsius
- Lineal
- ± 0.2 Precisión a 25°C
- Rango -100°C a 250°C
  Estabilidad a 100°C: 0.02°C/Año (Cerámica)

Figura 4.9 Sensor de temperatura de piel

Medición de oxígeno (PSR-11-917-MHJ) [38]

El caso de la medición de oxígeno puede ser uno de los más especiales, debido a la poca documentación de ciertos modelos que existen y el recelo de sus productores que inducen y en algunos casos obligan a los compradores al uso único de sus productos. Lo ideal en cualquier caso será, además de contar con un sensor para la medición de oxígeno en el ambiente, contar con un sensor biométrico para conocer el porcentaje de oxígeno en la sangre, y a partir de

dichos datos, tomar una decisión correcta de aumentar o disminuir el oxígeno provisto según la necesidad particular del infante. Una propuesta alterna para esto es el uso de un sensor de bajo coste como lo es el sensor Grove O^2 [39], el cual cumple la necesidad básica para el control de la provisión de la variable a controlar, en conjunto al sensor PLUX para la correcta retroalimentación del sistema [40]. Sin embargo, debido a la naturaleza del proyecto se decidió utilizar el sensor PSR-11-917-MHJ, de uso médico, para la medición del oxígeno del ambiente, generando una situación más real a la que los equipos médicos que se pueden encontrar en el mercado. Este sensor serviría como ejemplo para demostrar el propósito de crear un controlador configurable y débilmente acoplado, ya que, al igual que otros sensores de la industria médica, requieren de un cambio de filtros cada dos meses (lo que muchos servicios de mantenimiento ya tienen contemplado para mantenimiento preventivo), y la propia vida del sensor en si es de aproximadamente dos años, claro que podrían funcionar un tiempo más, pero incluso el fabricante ya no asegura su correcto funcionamiento, lo cual es crítico al momento de hablar de equipos y sus elemento de este tipo, donde existen vidas en juego.

El sensor de oxígeno (Figura 4.10) comparte una conexión parecida a las dos anteriores, solo tiene dos terminales, una que va a tierra y otra de salida, y al otorgar dicha salida en mV no es necesario agregarle una resistencia. Pero a diferencia de contar con un conector (Jack) universal como el sensor de piel, este requiere de uno propio, o de la conexión directa con el *plug*. Algunas de sus características son:

- Rango de medición 0-100%
- Precisión ± 2%
- Tiempo de respuesta de ≤9S
- Rango de temperatura 0 a 45°C
- Salida Análoga
- Comportamiento lineal
- Duración filtro 4 meses
- Vida útil 2 años

Figura 4.10 Sensor de oxígeno

Medición de humedad (HIH4000) [41]

La medición de humedad de ambiente se podría considerar de diferente manera en comparación a las variables anteriores, debido a que por lo general dichos sensores (temperatura y oxigeno) no se encuentran por si solos si no es hasta la gama alta, y si se busca un sensor "sencillo", habrá que hacer uso de módulos que incluyan dicha variable. Por lo regular, esta es medida en conjunto a la temperatura, por lo que se propone el uso del sensor SHT15 el cual tiene una precisión mayor a otros sensores como por ejemplo al del sensor DHT11 [42] [43]. Es así, que se optó por un sensor de alto desempeño que permite medir la humedad relativa del ambiente en un espectro más amplio a los anteriores y con una mayor precisión. Tanto este sensor, como el de oxigeno requieren un tiempo de asentamiento a diferencia de los demás sensores, así como un periodo de muestreo mayor debido al tiempo de respuesta que presentan.

El sensor de humedad HIH4000 (Figura 4.11) cuenta con un funcionamiento y configuración muy parecida a la del LM35, constituido con tres pines (Vcc, Vout y GND), permitiendo la lectura de la humedad relativa en el ambiente de manera porcentual. Algunas de sus características son:

- Salida aproximadamente lineal
- Rango 0°C a 50°C
- ± 3.5% Precisión a 25°C
- Salida Análoga
- Rango 0-100% RH

— Consumo de corriente de 200 µA
— No recomendable para aplicaciones de periodos extendidos donde RH >90%

Figura 4.11 Sensor de humedad

Si bien se podría hacer uso de módulos como el CM-42950 MX [44], el cual permite la medición de distintas variables (dióxido de carbono, temperatura, presión barométrica y humedad relativa) o de algún otro de este tipo; por recomendación del fabricante en cuanto a su aplicación (existiendo sensores para el uso médico o general) y debido a la naturaleza propia del controlador a desarrollar se optó por utilizar sensores dedicados para cada variable y especializados para así apegarse más a los estándares requeridos en la industria médica en cuanto a este tipo de equipos de soporte de vida se refiere, así como para presentarnos una situación más real a la que el controlador se podría enfrentar.

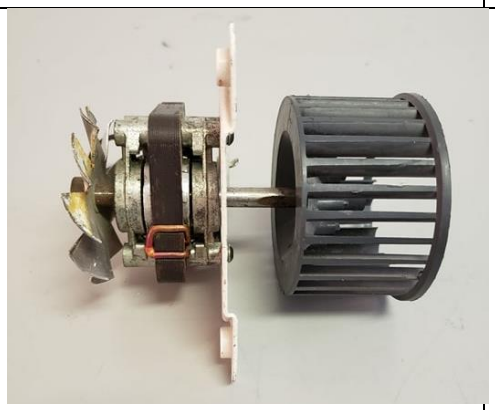
4.6 Actuadores

Al igual que en el caso de los sensores no existe una definición que permita acotar el número y tipo de actuadores a utilizar en una incubadora, y por lo tanto se carece de una configuración donde se determinen los sensores/actuadores a utilizar. En el sistema únicamente se contempla el uso de un actuador para el control de temperatura (compuesto de los dos elementos, resistencia calefactora y ventilador) y de una etapa de potencia dirigida a una aplicación casi particular (alimentar correctamente al actuador con una señal AC), sin embargo, la estructura de acoplamiento débil permitirá agregar cualquier otra cantidad y funciones dentro de los límites físicos de las tarjetas de control propuesto.

Control de temperatura

Para el control de temperatura se emplea una resistencia calefactora en conjunto con un ventilador para mejorar la propagación de calor. Las características de ambos se pueden apreciar en la Tabla 4.1, donde se incluyen el ventilador (Figura 4.12) y la resistencia calefactora (Figura 4.13):

Tabla 4.1 Elementos para el control de Temperatura

Ventilador	**Resistencia calefactora**
• Motor Mc Millan A264 • Voltaje de operación: 120V • Consumo: 0.23A • Potencia: 1/900 HP	• Voltaje de operación: 120V • Consumo: 2A
 Figura 4.12 Ventilador	 Figura 4.13 Resistencia calefactora

La temperatura es una variable que se puede considerar “lenta”, debido a que el incremento (o decremento) de esta sucede gradualmente y no cambia de un nivel a otro de una manera instantánea (además, no sería una situación deseable para cualquier ser vivo), esto otorga una gran posibilidad de controlarla, permitiendo el uso de diferentes métodos de control (ejemplo, controles On/Off, control proporcional, integral, derivativo, entre otros), por lo que la potencia proporcionada controlará la potencia otorgada en función de la diferencia (o error) que existe entre la temperatura actual y la deseada (o de referencia).

En el control de temperatura el actuador es la resistencia calefactora, y el ventilador únicamente sirve para la distribución uniforme del incremento de temperatura creado por la resistencia. Es así que para el control del motor del ventilador se emplea en el modo encendido / apagado, el cual se enciende una vez lograda la temperatura deseada para poder distribuir el aire cercano a la resistencia a lo largo del habitáculo de la incubadora, y es apagado después de un tiempo. En cambio para la resistencia calefactora se implementó un control tipo proporcional. La interfaz de potencia fue implementada con base en triac, un detector de cruce por cero y el control del ángulo de disparo. Figuras 4.14, 415 y 4.16 respectivamente.

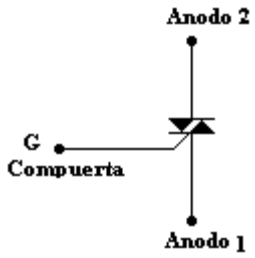


Figura 4.14 Construcción básica de Triac

En las gráficas de la Figura 4.15, se muestra tanto la señal de disparo del tiristor con la señal de alimentación de la carga con respecto al tiempo. La primer muestra el Va, el cual es el pulso de voltaje en la compuerta del triac para disparar el tiristor y aplicar corriente a la carga (resistencia) como se puede observar en la señal de la segunda gráfica de la figura. La señal de alimentación como la de control requieren estar en sincronizadas, por lo que se hace uso de un circuito para identificar el cruce por cero del voltaje de alimentación de la resistencia (circuito detector de cruce por cero).

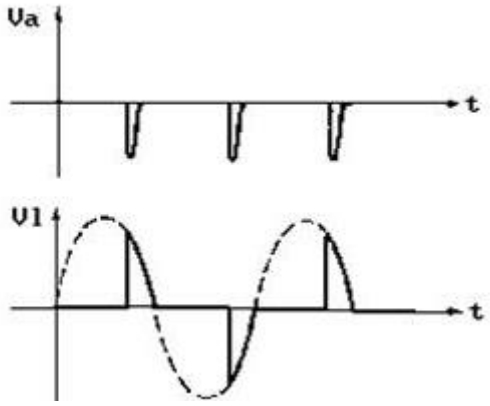


Figura 4.15 Señales de control cruce por cero

El control por del ángulo de disparo es bastante flexible ya que permite posible controlar la energía suministrada al actuador, en este caso, a la resistencia. Un circuito de control de este tipo, en el cual se puede seleccionar el ángulo de disparo, y por lo tanto el porcentaje de "trabajo" o conducción de la alimentación de la carga aporta flexibilidad para la utilización de diferentes técnicas de control.

Circuito de detección de cruce por cero

Como su nombre lo indica y se mencionó anteriormente, el circuito de detección de cruce por cero permite sincronizar la señal de la alimentación de la carga con la señal de control y así poder manejar correctamente el disparo de los tiristores. En cuanto a los detalles de diseño, se decidió crear un circuito para onda completa y así aprovechar la señal de AC completa (parte positiva y negativa), por lo que se inició rectificando la onda con un arreglo de cuatro diodos, para posteriormente enviar la señal a la entrada del opto acoplador, en este caso un 4n25, el cual tiene en la entrada un fotodiodo y a la salida un transistor, el cual al ser excitada su base conduce corriente de su colector al emisor, el cual está a 5V, y se conecta a una de las entradas del sistema. Para indicar el cruce por cero del voltaje de alimentación, Figura 4.16.

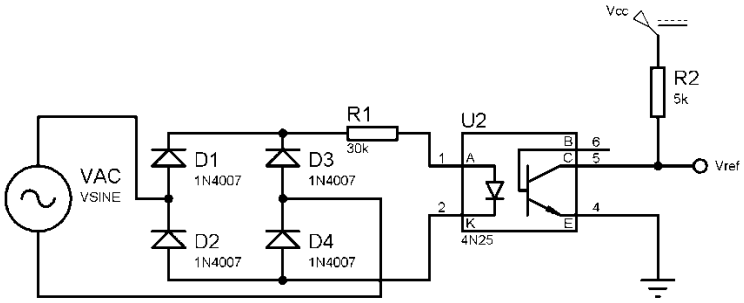


Figura 4.16 Circuito de detección de cruce por cero

Circuito de disparo

Una vez obtenida la señal que permite sincronizar la señal de alimentación y la señal de control, el siguiente paso es controlar una carga de AC con un consumo de 180W. Para tal efecto se empleó el circuito mostrado en la Figura 4.17. Se emplea una etapa de asilamiento (otorgada por el opto acoplador) para así no dañar algún componente en caso de tener alguna señal o ruido de la carga que regrese al sistema de control. Para esto se utilizó el MOC3021 el cual tiene a la entrada un foto diodo y a la salida un triac, dicha salida es utilizada para controlar al tiristor. Mediante el Triac (U3) es posible controlar el voltaje aplicado a la carga, mediante el control de ángulo de disparo. El tiristor empleado es un triac MAC16D, el cual permite una corriente de 16A.

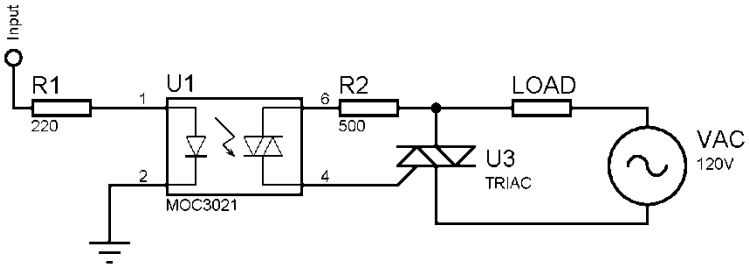


Figura 4.17 Circuito de disparo de tiristor

A continuación se muestra el circuito de la etapa de potencia (Figura 4.18) implementado en una tablilla de desarrollo, incluyendo las dos etapas antes mencionadas.

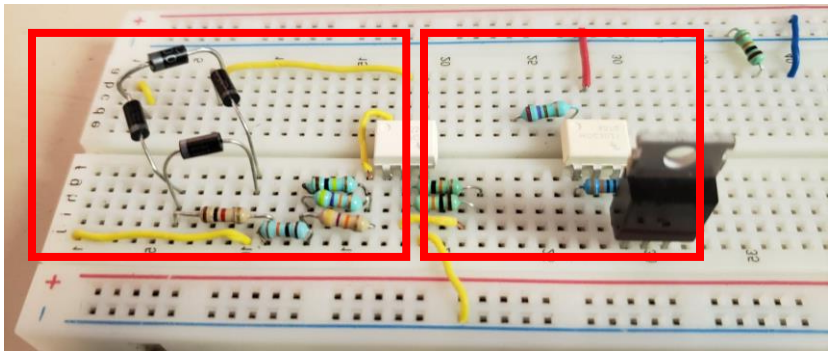


Figura 4.18 Etapa de potencia implementada

4.7 Lectura / Escritura de Datos.

El sistema basado en Windows IoT opera la tarjeta Raspberry para el manejo de los recursos del equipo y realizar la lógica del sistema. Así mismo, permite realizar la función de configuración del sistema, como es la identificación del sensor a leer por el Arduino, conversión de los valores medidos a deseados, impresión de los valores en la página principal para la lectura del personal y el ajuste de la señal de control. Todo esto depende de los parámetros configurados por el operador del equipo. Es importante resaltar que al inicio de la operación el sistema sólo cuenta con la estructura base la cual hace uso de colecciones que contendrán los objetos Sensor, Actuador y Controlador definidos (configurados) por el usuario técnico, donde cada objeto cuenta con parámetros diferentes para su utilización como lo son descritos en el punto 4.4.

El Arduino como antes se ha mencionado, es utilizado en conjunto con la Raspberry Pi a través del módulo desarrollado para su comunicación utilizando el protocolo I^2C, contando así con la lógica necesaria para funcionar como esclavo de la tarjeta Raspberry en el esquema Maestro-Esclavo. El código cíclico del Arduino al recibir una orden de lectura, procede a leer el sensor solicitado y enviar su valor a la Raspberry. Igualmente para el caso de las señales de control enviadas por la Raspberry, al recibirla se tiene información de puerto de salida y su

valor. Cabe mencionar que dichos puertos y protocolos de comunicación se encuentran únicamente limitados por la disposición de incorporar el modulo correspondiente para su utilización añadiendo así las particularidades deseadas, sin requerir cambio de código debido a que la estructura seguirá siendo la misma, al igual que cuando se agregan nuevos sensores y actuadores.

Ordenes de lectura

El Arduino al contar con entradas analógica permite leer los valores otorgados por los sensores haciendo la función de ADC, de los cuales, en casos particulares como el del sensor de temperatura cutánea se requiere de un acondicionamiento previo a la entrada analógica para su correcta medición (esto debido a la señal tan pequeña que proporciona). Para dicha medición se utilizan las entradas A0 – A3 del Arduino (el A4 y A5 igualmente pueden ser utilizados si el Arduino permite la conexión I^2C a través de otros pines, asimismo hay que recordar que el Arduino UNO solo es de referencia para este caso en particular, pero es posible el uso de cualquier dispositivo capaz de comunicarse por I^2C, siempre y cuando cumpla con las necesidades de velocidad y resolución de lectura y escritura de datos). Del proceso de lectura de variables se obtendrán dos salidas principales, la primera que es para el manejo del sistema de alarmas físico, y el segundo el cual es el envío de los datos medidos hacia la Raspberry para su posterior procesamiento. Posteriormente, sirve para obtener la salida a la etapa de potencia para finalmente controlar los actuadores a través de los pines PWM con los que cuenta (D3 – D7 y D9 – D11).

Ordenes de escritura

En cuanto a las salidas de la tarjeta, de manera ilustrativa se decidió utilizar el PWM del Arduino (pin 12) como salida optativa para el actuador que requiera un control de este tipo. Mientras que para el resto de actuadores se dejó indicado los demás pines como si de un control On-Off fuera, cuyo diseño se podrá ser modificado para hacer un control más complejo al utilizar los demás pines GPIOs y así obtener salidas con una mayor resolución, o simplemente contar con más PWMs. Finalmente a través de los drivers de potencia para cada actuador se tendrá el control de las variables.

La etapa de potencia se representa como una caja negra o bloque genérico en el diagrama estructural del sistema (Figura 4.4 para referirse a ella como una etapa general para el control de todos los actuadores para facilitar el entendimiento del esquemático, sin

embargo, dicha etapa será independiente según los actuadores y las señales de salida obtenidas de la Raspberry en su implementación.

Ciertos sensores analógicos producen una señal muy pequeña por lo que requieren una etapa de acondicionamiento de señal. Así mismo la señal de salida del sistema requiere una etapa de potencia para manejar el actuador. El análisis y solución del problema de acondicionamiento de señales pudiera ser muy amplio, lo cual queda fuera del tema central del presente proyecto. Por lo tanto, se plantean sistemas de acondicionamiento de uso común para lograr verificar la operación del sistema informático utilizando sensores y actuadores reales.

4.7.1. Lectura de datos

La comunicación Raspberry-Arduino se implementó mediante el protocolo I^2C, por su soporte nativo para la mayoría de dispositivos USB en la plataforma de Windows IoT. Así mismo se facilita el manejo de los elementos del sistema asignando una dirección (o tag) para el llamado a los distintos sensores y dispositivos, de manera maestro-esclavo. El maestro (Raspberry) cuestiona según las necesidades del sistema los valores a medir al esclavo (Arduino). Es importante destacar que así como se utiliza la tarjeta Arduino, es posible utilizar cualquier otra tarjeta que soporte el protocolo I2C. Es deseable utilizar protocolo que requiera una dirección o ID para referirse al elemento. Ya que de esta manera se puede estandarizar la comunicación de dispositivos de una manera Request-Receive debido a que su flexibilidad se encuentra basada en saber únicamente la dirección del elemento y si se tiene que realizar una acción de lectura o escritura permitiendo la flexibilidad en este punto donde no solo se tengan que tener dichos campos para lograr la comunicación. Algo parecido a lo antes mencionado en cuanto a la rutina de lectura, conversión y cálculo de la señal de control del lanzador de órdenes el cual solo es necesario hacer uso de “ordenes” generales para controlar las funciones particulares de cada objeto.

4.7.2 Comunicación I^2C

Como se ha mencionado el uso del protocolo I^2C requiere mínimo de dos partes que trabajen en conjunto para la correcta comunicación entre ellos, a continuación se describe la configuración para ambas partes:

Raspberry (Maestro)

Para utilizar el protocolo I^2C (en Windows IoT) se debe de utilizar el espacio de nombres “Windows.Devices.I2c” el cual provee acceso a dicho modulo del sistema. Con el cual es posible crear una instancia de dispositivo a conectar como esclavo y sus configuraciones respectivas tales como velocidad del bus y su dirección.

La lectura de los sensores se implementó a manera de sondeo (polling) mediante temporizador se cuestionará al dispositivo lector de los sensores, cambiando de entrada en cada ciclo, actualizando así mismo el valor de dichas variables; tomando en cuenta el tiempo mínimo necesario para el muestreo de las variables de cada sensor.

Arduino (Esclavo)

En el lado esclavo de la comunicación se encuentra una placa Arduino, haciendo uso de la librería “Wire.h”; se define la dirección del dispositivo y se manejan los valores a enviar según los requests dados por el master.

La rutina base del Arduino se enfoca en la medición de los sensores y el envío de los valores medidos requeridos por la Raspberry, este hace uso de las funciones millis que controlan el uso del timer de la plataforma para estar midiendo los valores analógicos de los sensores.

4.7.3 Configuración de sensores

Inicialmente el sistema no tiene un conjunto de objetos sensor o actuador, previamente definido. El usuario-técnico es el encargado de configurarlos. El sistema implementado hace uso de cuatro sensores diferentes, a los cuales se les asignó la dirección 0xE3, 0xE4, 0xE5 y 0xE6, las cuales para el momento de configurar algún sensor nuevo deberán ser configuradas por el usuario de acuerdo a la dirección real (ADC, tarjeta de adquisición de datos, etc...) en la pantalla correspondiente de la interfaz gráfica. Estos sensores son analógicos y otorgan valores en mV (a excepción de termistor el cual otorga valores resistivos, como se desarrolla a continuación), por lo cual, y debido a las especificaciones de las normas en cuanto a la precisión de las variables, utilizar el ADC (en forma de entradas analógicas) del Arduino de 10 bits nos presenta una plataforma más que funcional para su aplicación.

4.7.4 Proceso de lectura y escritura de datos

En esta sección se explica de una manera breve el funcionamiento básico del sistema, y la comunicación de sus elementos. Una vez establecida la conexión entre el sistema base (Raspberry/maestro) y la tarjeta de adquisición de datos (en este caso el Arduino/esclavo) el maestro envía las direcciones de los sensores que se desean medir. Se cuenta con un Buffer para almacenar temporalmente los valores recibidos hasta la siguiente iteración de solicitud, dicha variable será enviada mediante una función asíncrona localizada en la clase Control (lanzador de ordenes) en donde se convertirán los valores según el tipo de sensor que se tenga, para posteriormente ser almacenados en las variables locales respectivas de cada sensor, y actualizar su valor en la interfaz gráfica creada relacionando las variables con sus respectivos elementos en XAML.

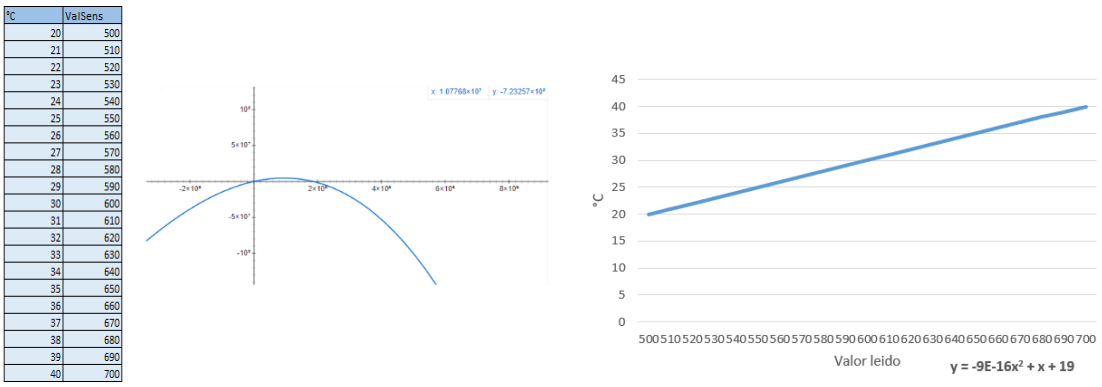
Debido a que la plataforma de Arduino no permite acceder directamente a los Timers del microcontrolador Atmega328, se tiene en la necesidad de utilizar las funciones proporcionadas por los mismos desarrolladores de Arduino para el manejo de funciones de tiempo, como es el caso de la función milis(), la cual regresa el número de milisegundos que han pasado desde que se inició la tarjeta. Con dicha cuenta es que se creó el ciclo de medición de los sensores conectados a las entradas análogas (dejando un tiempo entre medición y medición para la correcta lectura de los sensores). Esto se encuentra en el loop principal del Arduino. Mientras tanto con uso de la librería “Wire”, la tarjeta se encuentra a la espera “onReceive(receiveEvent)” de alguna dirección (enviada por la Raspberry). Al momento de recibir la solicitud de un elemento “onRequest(requestEvent)” utilizamos la función “Write” para enviar el valor de la variable leída en el ciclo antes mencionado.

Conversión de las mediciones obtenidas por los sensores

En el proceso de configuración, el usuario-técnico introduce los distintos parámetros necesarios para la creación del objeto Sensor, entre los cuales se encuentran los factores de un polinomio de segundo grado el cual a partir de dicha ecuación caracteriza al comportamiento del sensor en particular, llevando a realizar la correcta conversión entre los valores leídos por el ADC a los valores en las unidades medición en este caso de temperatura, deseadas por el usuario.

En el ejemplo de la figura 4.23, se muestra la tabla que representa la relación entre los grados centígrados (unidades deseadas) y el valor leído a través de la entrada analógica del Arduino. A partir de la tabla es posible definir un polinomio de segundo orden que permita describir el comportamiento del sensor en particular. Las gráficas que se muestran en la figura 4.23, representan la misma ecuación de segundo grado. En la gráfica de la izquierda se representa mayor intervalo de valores (rango) y en la gráfica de la derecha sólo se grafican los puntos en el intervalo de interés y se puede observar que al igual que como lo especifica el fabricante del sensor LM35 su comportamiento para el rango de uso es casi lineal. En caso de requerir mayor precisión es necesario utilizar una ecuación de mayor orden ya sea para el caso de la utilización de sensores diferentes o que requieran de una lectura con una resolución mayor. La conversiones de los valores leídos a los valores en las unidades deseadas a través la ecuación polinómica de segundo orden creada definida por el usuario es fácilmente llevada a cabo por la tarjeta Raspberry, ya que en el caso de contar con un sistema más básico (en cuanto a hardware y poder de procesamiento) se tendrían que tener mayores consideraciones en cuanto al manejo de recursos de la tarjeta y tiempos de procesamiento.

Un polinomio de segundo grado como lo es: $f(x) = ax^2 + bx + c$ cuenta con tres parámetros que multiplican a la variable (valor leído x) a, b y una constante c, ofreciéndonos una ecuación que permite obtener la relación del valor en las unidades deseadas y del valor leído y obtenido a través del Arduino. En el caso del sensor aludido en el párrafo anterior si se obtiene una medición de 500 (x), al sustituirlo en el polinomio característico con parámetros $a = -9e^{-16}$, $b = 1$ y $c = 19$, obtendiarmos un valor de 20°C. La grafica característica de estos parámetros para el elemento de ejemplo se muestra en la Figura 4.19.



°C	ValSens
20	500
21	510
22	520
23	530
24	540
25	550
26	560
27	570
28	580
29	590
30	600
31	610
32	620
33	630
34	640
35	650
36	660
37	670
38	680
39	690
40	700

Figura 4.19 Caracterización sensor temperatura

4.8 Plataforma computacional

En la presente sección se describe el desarrollo del sistema informático empezando por la selección del sistema Windows IoT para el manejo de recursos de la tarjeta, sus ventajas y las posibilidades que presenta en comparación con otros sistemas operativos, así como la configuración del mismo, la creación del Framework, la integración de las demás partes (lectura y conversión de valores), la estructura dinámica de presentación de valores medidos en la pantalla de usuario y el sistema de páginas para el manejo de la interfaz gráfica.

4.8.1 Sistema operativo

La Raspberry Pi, así como cualquier otra tarjeta de desarrollo, requiere de algún tipo de software o sistema operativo para su funcionamiento, para tener acceso a su hardware desde un alto nivel. Actualmente se cuenta con varias opciones para el manejo de los recursos de la tarjeta, entre ellas: las más conocida Raspbian (Raspberry OS) basada en Debian (Linux), y Windows IoT Core, la solución propuesta por Microsoft específicamente para el desarrollo de proyectos basados en el "Internet of Things"; las cuales dado su enfoque, documentación y antecedentes en el desarrollo de proyectos y aplicaciones de sistemas embebidos se presentan como una opción atractiva para llevar a cabo tanto el desarrollo del controlador de la incubadora como de la interfaz gráfica. A continuación se describen las principales características de dichas propuestas tomadas como base para seleccionar la más acorde a las necesidades del sistema que se pretende realizar.

Raspbian

Sistema operativo completo, el cual puede trabajarse independientemente (Standalone), y está basado en Debian (Linux). Incorpora herramientas de desarrollo de aplicaciones y manejo de puertos pre-instalados, siendo Python el principal lenguaje utilizado para el desarrollo de aplicaciones a través de entorno de desarrollo IDLE. Es el sistema operativo oficial de la Raspberry Pi, en la Figura 4.20 se muestra el logo que identifica a este sistema operativo; Raspbian cuenta con una amplia comunidad, la cual tiene años de experiencia otorgando ayuda y soporte de manera gratuita. Esta a su vez se divide en dos versiones Raspbian Pixel y Raspbian Lite, siendo la primera la versión "completa" ofreciendo la experiencia de una computadora de sobremesa, y la segunda una versión reducida sin entorno gráfico, utilizada comúnmente para servidores [45].

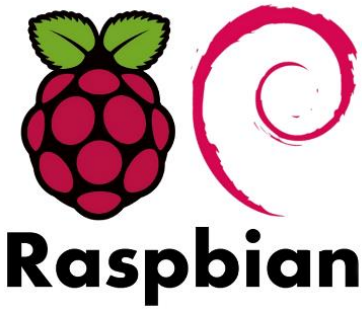


Figura 4.20 Logo Raspbian

Windows 10 IoT Core

Versión "simplificada" de Windows 10, su logo se muestra en la Figura 4.21. Este sistema solo permite correr una aplicación en primer plano, y requiere de una PC con Windows 10 para el desarrollo de sus aplicaciones. A pesar de su aparentes limitaciones, el seguir haciendo uso del núcleo de Windows permite desarrollar aplicaciones a través de herramientas como por ejemplo Visual Studio, tanto en C# como Visual Basic .Net, siendo el primer lenguaje el más utilizado para su programación. Si bien existe una versión "Enterprise" del sistema operativo, la versión "Core" está diseñada especialmente para el uso en CPUs ARM. Además, a diferencia de Raspbian, Windows 10 IoT Core es una plataforma relativamente nueva, en donde no existe una comunidad tan grande y experimentada como lo es en Raspbian, no obstante, su utilización ha ido en aumento en tiempos recientes [46].

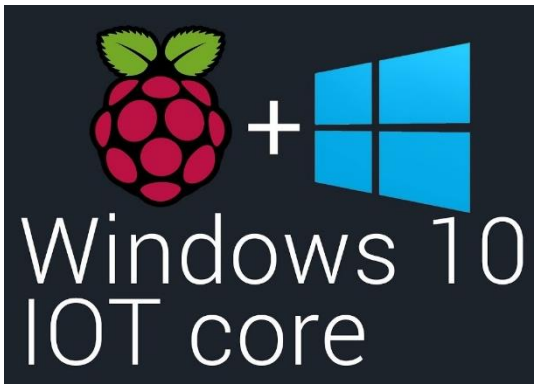


Figura 4.21 Logo Windows IoT

La flexibilidad que representa el uso de Raspbian sobre Windows 10 IoT Core es innegable, sin embargo, en el desarrollo de proyectos de investigación uno de los principales factores de importancia es la innovación, por lo que hacer uso de una plataforma aun no tan explorada, podría servir de ejemplo para proyectos futuro en especial para un área como lo es la de control flexible, además de ser un sistema operativo con un enfoque más centrado a la creación de proyectos y sistemas embebidos, y no tan general como lo es Raspbian. Otro aspecto de interés es el uso de las herramientas brindadas por Microsoft, el cual al habilitar herramientas de desarrollo más conocidas permitirá precisamente una mayor flexibilidad en el sistema. Cuenta con un enfoque de desarrollo para el Internet de las cosas, que incluye herramientas para la interacción del dispositivo con la nube de manera nativa en el entorno de desarrollo de Visual Studio.

4.9 Plataforma Windows IoT.

Existen algunos requisitos para el desarrollo de aplicaciones en Windows IoT a través de Visual Studio. Antes de iniciar a trabajar con Windows 10 IoT se requiere del uso de las herramientas de desarrollo de Windows 10 SDK, las cuales se pueden obtener a través de la página oficial de Microsoft [47], así como de habilitar el “Modo de programador” en los ajustes de Windows y encontrarse en la versión más reciente o deseada a trabajar de Windows 10.

4.9.1 Configuración Visual Studio 2017

Una vez abierto el entorno de Visual Studio, se crea un nuevo proyecto como si de cualquier otro se tratase, primero escogiendo el lenguaje a utilizar, y seleccionar Aplicación Vacía (Windows universal), seleccionando en la parte inferior derecha Crear nuevo repositorio GIT, la Figura 4.22 muestra dichas configuraciones.

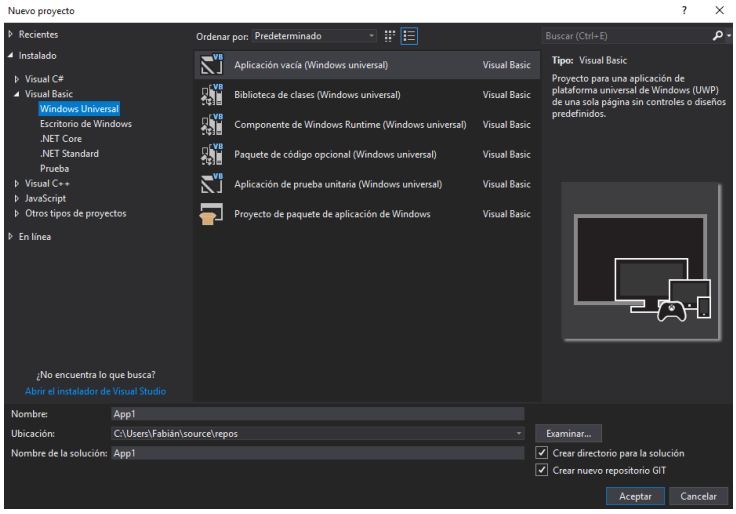


Figura 4.22 Pantalla Nuevo Proyecto

Luego se deberá agregar la referencia de Windows IoT Extensions for the UWP (última versión o la versión con la que se desee trabajar y haya seleccionado al inicio del proyecto), con lo que el entorno se encuentra ya preparado para iniciar a desarrollar la aplicación como normalmente se realiza en Visual Studio.

4.9.2 Universal Windows Plataform y XAML

UWP es la plataforma y núcleo de Windows, lo que permite desarrollar aplicaciones para distintos dispositivos que utilicen dicho sistema operativo, de una manera flexible, y adaptativa. La cual es programable en C#, C++, Visual Basic y Javascript. Permitiendo desarrollar la UI (interfaz de usuario) en XAML, HTML o DirectX. Para el desarrollo de este proyecto se decidió utilizar Visual Basic como lenguaje de desarrollo del controlador, y XAML para la interfaz gráfica.

4.9.3 Interfaz de usuario

El sistema requiere de una interfaz que permita al usuario tener contacto con el equipo, desde simple lectura de valores hasta la interacción y configuración de sus parámetros a través de la misma. A continuación se detalla el camino tomado para atacar la necesidad de tener una interfaz amigable, funcional y de fácil acceso para el usuario.

Interfaz preliminar

Con el propósito de realizar ciertas pruebas de la plataforma, se creó una plantilla de interfaz gráfica, en formato .JPG y fue añadida al proyecto en el directorio "Assets" de la carpeta del proyecto (Figura 4.23). En la cual se pudieran observar una cantidad de sensores o campos limitada debido a los parámetros prestablecidos de temperatura ambiente, temperatura de piel, humedad, y oxígeno, dividida en secciones según el tipo de elemento para facilitar su lectura y comprobar la rutina de lectura y conversión de valores leídos de los sensores. Esto va en contra del mismo propósito del trabajo de tesis, sin embargo sirvió para las pruebas de correcta lectura de valores. Una vez verificado el correcto funcionamiento del sistema, se eliminó dicha plantilla para continuar con el siguiente paso lógico, una interfaz gráfica con la que tiene contacto el personal directo del hospital y las variables medidas.

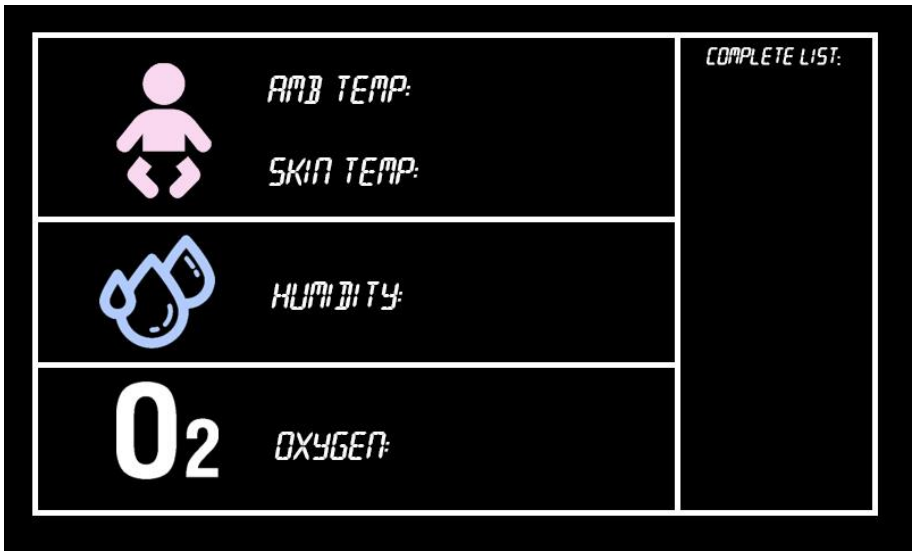


Figura 4.23 Plantilla de interfaz gráfica de prueba

Debido a la naturaleza propia del proyecto en donde se desconoce el número de sensores y actuadores se optó por emplear en lugar de una plantilla estática o un espacio designado limitado, emplear la herramienta de ListView, la cual permite mostrar tantos elementos en la interfaz de usuario como objetos de una colección existan, teniendo acceso directo a las propiedades de estos, como el nombre, ID y valor medido facilitando y estandarizando su despliegue en la pantalla de lectura.

Esto forma parte de los esfuerzos de la plataforma, la cual además permite crear interfaces "genéricas", que se adaptan por si solas a diferentes resoluciones que puedan tener los dispositivos en donde se quieran implementar, justamente con el mismo enfoque de flexibilidad en el que está basado el proyecto.

Estructura de datos ListView

La Figura 4.24 presenta el bloque de los las propiedades que conforman el campo del elemento en Listview a instanciar, así mismo se puede observar cómo se utiliza la propiedad Binding de la herramienta para acceder a las propiedades del sensor y mostrarlas en dentro de textblocks para así formar la estructura mostrada en la Figura 4.24: ID, Nombre, Valor medido y Unidades.

```
<ListView x:Name="ListSens" HorizontalAlignment="Center" Height="400" VerticalAlignment="Center" Width="780">
    <ListView.ItemTemplate>
        <DataTemplate>
            <StackPanel Orientation="Horizontal" Padding="5">
                <TextBlock Text="{Binding ID}" Margin="20,0,20,8" FontSize="60" FontWeight="SemiBold" Foreground="White"/>
                <TextBlock Text="{Binding Nombre}" Margin="20,0,20,8" FontSize="60" FontWeight="SemiBold" Foreground="White"/>
                <TextBlock Text="{Binding sRValue}" Margin="20,0,20,8" FontSize="60" FontWeight="SemiBold" Foreground="White"/>
                <TextBlock Text="{Binding Unidades}" Margin="20,0,20,8" FontSize="60" FontWeight="SemiBold" Foreground="White"/>
            </StackPanel>
        </DataTemplate>
    </ListView.ItemTemplate>
</ListView>
```

Figura 4.24 Estructura ListView

Al ahondar en el entorno de desarrollo, en la Figura 4.25 se evidencia la ventana de Visual Studio donde desde esta se realiza la programación de aplicaciones para Windows IoT como lo podría ser para cualquier otra aplicación de la plataforma de Windows y los lenguajes que soporta el entorno. En la parte central se tiene una visualización de la pantalla o archivo XAML seleccionado, debajo de esta se cuenta con su código desde el cual se pueden administrar todos los elementos de la pantalla. A la derecha se encuentra el explorador de soluciones, donde en este caso se tiene seleccionado el archivo de la Pagina 1 para su edición.

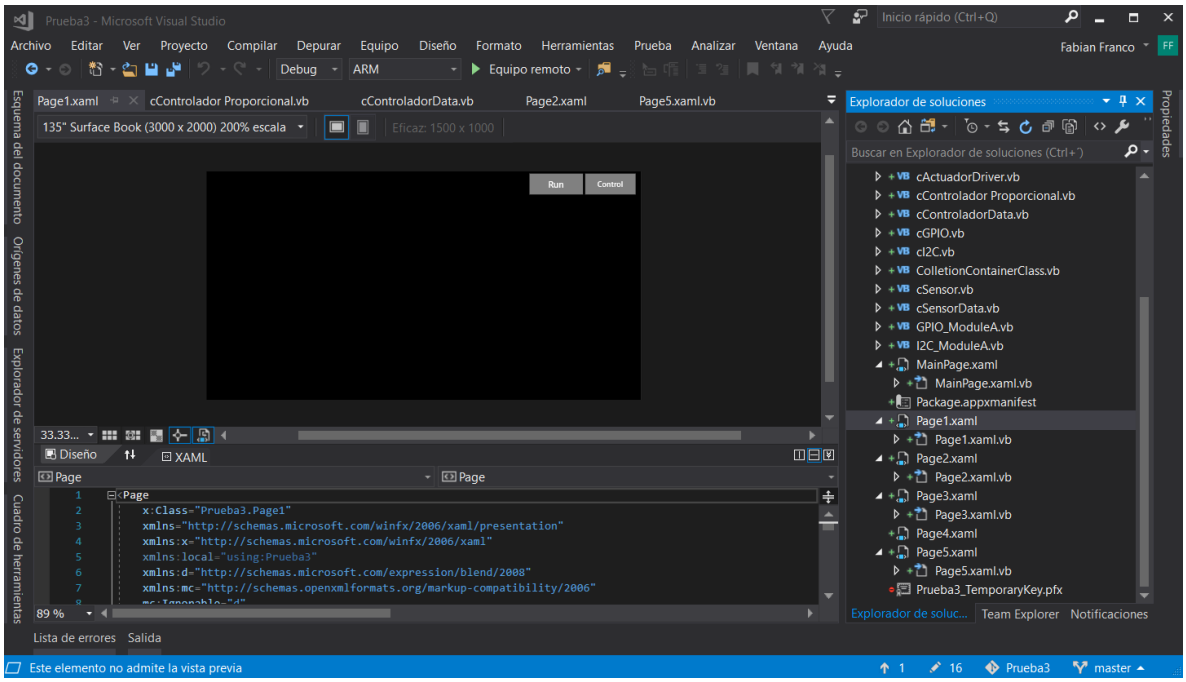


Figura 4.25 Página Principal Proyecto

Barra de navegación

En cuanto a la configuración para la pantalla, fue seleccionado un dispositivo TFT de 7” para utilizar en el sistema y mostrar la interfaz con la que tendrá interacción el usuario, esta cuenta con una resolución de 800x480p, la cual es configurada en la página principal la cual contiene el Frame de despliegue de las demás pantallas, por lo que se designó una resolución de 800x420p, donde 60p de ancho corresponden a una barra de navegación entre páginas. Esto no afecta a pantallas con diferente tamaño o resolución (ya que el propio sistema escala al formato indicado automáticamente), si no para destinar un espacio para la propia barra de navegación (Figura 4.26).

Figura 4.26 Barra de navegación entre páginas

En esta barra se cuenta con los accesos que dirigen a las pantallas de información, lectura de variables, creación de sensor y actuador como más adelante se explicara con mayor profundidad.

Debido a que las interfaces graficas en UWP son integradas en una sola ventana se creó un sistema de páginas en donde se tiene una página principal (MainPage.xaml), la cual cuenta con la resolución de pantalla seleccionada para la implementación del proyecto (la cual puede ser escalada automáticamente al conectar una pantalla externa a través del puerto HDMI) y botones de navegación a las distintas páginas a través de la barra de navegación. Estas contarán con un tamaño menor a la principal debido a que serán mostradas en el Frame creado en dicha página, teniendo así una resolución nativa de 800x420 pixeles a diferencia de la original de 800x480. A continuación se mostraran las distintas pantallas que conforman la interfaz de usuario.

En todas las pantallas, la barra de navegación se encuentra presente debido a que es a través de ella donde el usuario puede saltar de ventana a ventana en la interfaz unificada de UWP. A la izquierda de la Figura 4.28 se tiene a la pantalla de bienvenida, la cual es la primera en presentarse al usuario al iniciar el equipo, a partir de este punto el usuario seleccionara la pantalla a la cual desea dirigirse, en caso de dar clic en la “i” se mostrara la

pantalla derecha de la Figura 4.27, donde se muestra información básica acerca del equipo y sus funciones.

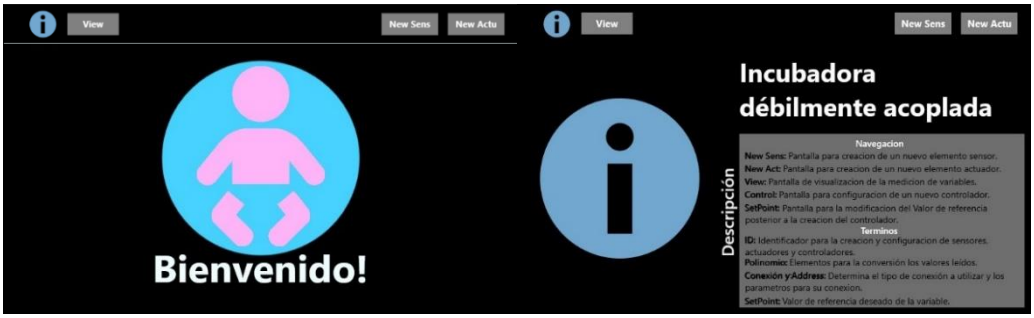


Figura 4.27 Pantalla de Bienvendida e Información

La pantalla de mayor utilidad, o al menos para el personal médico es la de lectura de datos (Figura 4.28), ya que en ella se pueden apreciar las mediciones de los sensores configurados para el equipo.

Figura 4.28 Pantalla de lectura de datos

Por otro lado, las pantallas de interés para el personal que será encargado de la configuración del equipo serán las mostradas en la Figura 4.29, donde se tienen la pantalla de configuración de sensores y actuadores, permitiendo la adición de los elementos con las características mostradas en la misma figura, controladores que permiten ligar a un sensor y a un actuador para el manejo de una variable o parámetro que se quiera controlar y de configuración del valor de referencia en caso de que se quiera modificar dicho valor posterior a la configuración de los elementos.

Figura 4.29 Pantallas de configuración

4.9.4 Visualización de mediciones en interfaz grafica

Una vez calculado el valor medido y convertido a las unidades deseadas, es momento de mostrarlo en la página de monitoreo de variables (Page1). Como se menciona en el punto 4.9.3 para el despliegue de valores se utiliza la herramienta ListView, la cual permite acceder a las propiedades de ID, Nombre, Valor y Unidades propias del objeto sensor instanciado y los despliega a través de Textblocks que integran al bloque de elementos de ListView mostrado creando así una línea por sensor con la estructura mostrada en la Figura 4.30.

Figura 4.30 Propiedades de elemento para su visualización

A manera de ejemplo, en caso de que el usuario decidiera asignar a las propiedades los valores listados en la Tabla 4.2, se obtendría como resultado una línea o campo como el presentado en la Figura 4.25.

Tabla 4.2 Ejemplo valor de propiedades

ID	1
Nombre	Temperatura de Piel
Valor	*Obtenido al convertir el valor medido
Unidades	°C

De acuerdo a la estructura de la Figura 4.31, primero muestra el ID del objeto, luego el nombre, el valor convertido ya en las unidades deseadas y las unidades indicadas a través de como al usuario se le facilite su lectura.

Figura 4.31 Ejemplo de elemento sensor mostrado en la interfaz

4.10 Desacoplamiento del sistema

Un aspecto del sistema Windows IoT Core de relevancia para la rigidez o flexibilidad del sistema, es la interfaz gráfica que permite únicamente el despliegue de una ventana a la vez. Situación que requiere un esfuerzo extra para lograr el desacoplamiento del sistema. A continuación se describirá el funcionamiento de la interfaz gráfica con la que el usuario tendrá interacción, el manejo de clases y la abstracción de propiedades.

4.10.1 Sistema de páginas

El sistema Windows IoT Core cuenta con una interfaz gráfica unificada, por lo cual una de sus "limitaciones" es el despliegue de una sola pantalla a la vez, a diferencia de Windows 10 donde se pueden tener varias ventanas abiertas y en pantalla al mismo tiempo. Las pantallas pueden ser definidas como elementos de interacción con el usuario y pueden estar dedicadas a funciones particulares, además de que permiten él envió de objetos entre diferentes pantallas, parte fundamental para el proyecto. El diagrama presentado en la Figura 4.32 muestra el direccionamiento de las distintas pantallas que conforman al sistema informático y fueron mencionadas en el pasado punto 4.4.3 y cuyas interfaces de usuario se pueden observar a través de las Figuras 4.27 a 4.29.

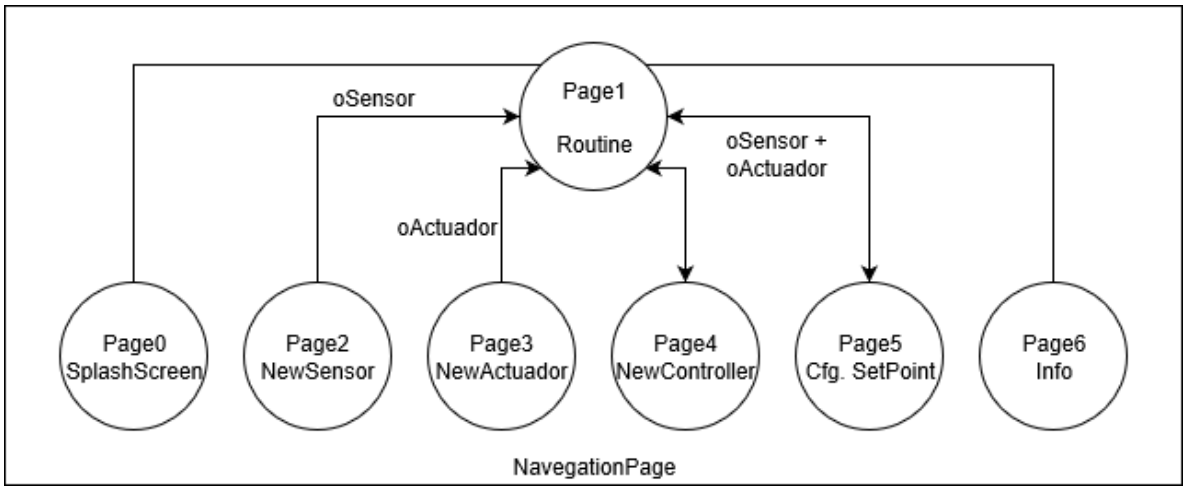


Figura 4.32 Direccionamiento de páginas

Descripción de las páginas

- **NavegationPage.** Funciona como interfaz de navegación entre las demás páginas, se encuentra presente en todo momento. Cuenta con dos botones que muestran las pantallas de Creación de sensor y creación de actuador respectivamente, un icono para dirigirse a la página de información y uno adicional para regresar a la página de lectura de variables (Page1).

- **Page0.** Pantalla de inicio, siendo la primera que el usuario es capaz de visualizar dentro del Frame al iniciar el equipo.

- **Page1.** Página de mayor interés para el personal del hospital, permite la lectura de los valores de las diferentes variables de una manera clara y sencilla. Estas son mostrados con la ayuda de ListView, el cual nos permite instanciar distintos elementos de lectura (relacionados a los objetos agregados a una colección) y agruparlos en forma de lista para tener una estructura con mayor orden en cuanto a identificación y referencia de los elementos.

- **Page2.** Para la configuración de sensores nuevos, en donde al usuario se le solicitará la localización del sensor en la primer página, el nombre y tipo de sensor a configurar, las unidades que leerá, el tipo de conexión a utilizar ($I^2C, ethernet$ ...), la dirección con la que se identificará el sensor en el protocolo de comunicación y los elementos de la ecuación característica (de hasta tercer grado) para la conversión del valor leído del sensor en las unidades deseadas (°C, °K, % HR, % Oxigeno, Kg).

- **Page3.** Configuración de las características del actuador, el usuario deberá establecer el ID con el que será reconocido y la dirección física del dispositivo.

- **Page4.** Proporciona una interfaz la cual le permite al usuario ligar cual actuador utilizar con el sensor deseado, así como la configuración propia del controlador, si es una señal de control de tipo digital o analógica, la ganancia para el tipo de control a implementar y el valor de referencia deseado al cual se desea controlar.

- **Page5.** Permite al usuario configurar el valor de referencia establecido al momento de la creación del controlador, para que de esta manera pueda ser actualizado en todo momento.

- **Page6.** Otorga una breve descripción del proyecto, sus capacidades y las propiedades necesarias para la configuración de los sensores y actuadores.

4.10.2 Desarrollo de las clases

Las clases forman una parte esencial en el paradigma orientado a objetos, ya que son las que nos permiten abstraer los datos y operaciones de diferentes tipos de objetos con características en común, consta de métodos y de datos que resumen las características comunes de dicho conjunto. Como se ha analizado en la estructura de ArquiTAM, así como en la estructura de clases Figura 4.5 Son las que nos permiten el desacoplamiento entre etapas y permiten la creación del Framework desarrollado, a continuación se detallan cada una de ellas y se presenta una parte de su código funcional para mayor entendimiento del sistema informático del sistema.

Clase control

Las figuras 4.5 y 4.6 ilustran las distintas clases y su relación. La clase Control, identificada como un lanzador de órdenes, cuenta con las rutinas de inicialización de los módulos de comunicación, la rutina de lectura de sensores, conversión de valores, despliegue de valores en pantalla y obtención de señal de control. A continuación en la tabla 4.3 se presenta un extracto del código de relevancia de esta clase:

Tabla 4.3 Extracto de código de la clase control

```
       Dim oSensor As cSensor
       Dim oActuador As cActuador
       Dim cSensors as collection
       Dim cActuadors as collection
       Dim cControladors as collection
```

```
       Public Sub Main()
           InitCom()
           TimerTick()
       End Sub
```

```
       Private Sub Timer_Tick()
               periodicTimer=NewTimer(AddressOf
PollingObjects&Control, Nothing, 0, 1500)
        End Sub
```

```
Public Sub PollingObjects()
        For Each item In cSensors
            item.ReadSens()
        Next item
        For Each item In cControladors
            item.Controlar()
        Next item
        Public Sub UptadeValues()
End Sub
```

```
Public Sub PageObjects()
      cSensors.Add = oSensor
      cActuadors.Add = oActuador
      cControls.Add = oControlador
End Sub
```

Como se puede observar el código se divide en distintas secciones, primero se declaran los objetos sensor, actuador y controlador como del tipo perteneciente a su respectiva clase, luego se inicializan los módulos de comunicación con InitCom y después empieza la rutina del timer, con cada Tick del timer declarado se hace una iteración de la subrutina, primero es el análisis de los sensores, se busca al primer elemento sensor de la colección, y se utiliza el comando general ReadSens para obtener el valor obtenido del sensor. Hay que recordar que las propiedades del objeto Sensor, Actuador y Controlador son especificadas por el usuario al momento de su configuración.

Una vez obtenido el valor es convertido a unidades por una función de la clase intermedia envoltura. Y es momento de verificar a los controladores, en dicha clase se configuro la acción de control a implementar por lo que se utiliza el comando genérico “Controlar” para realizar la acción de control a través del actuador ligado a este sensor y controlador. Una vez realizado esto, se actualizan los valores en la pantalla principal y se procede al siguiente elemento sensor de la colección.

Luego se tiene la parte de manejo de objetos, los cuales son recibidos una vez generados por el usuario en las correspondientes pantallas de configuración y posteriormente añadidos a su debida colección.

En esta parte se describirán las clases creadas con base en el mecanismo de acoplamiento débil (ArquiTam), empezando primero por la clase representante Sensor, la cual es muy parecida a la clase Actuador debido a su misma función de ser la clase representante para este tipo de elementos.

Clase representante

A continuación la Tabla 4.4 muestra el extracto de código de la clase Representante Sensor.

Tabla 4.4 Extracto de código de la clase representante Sensor

```
      Private Dispositivo As cSensorEnvoltura
```

```
      Public Sub New()
             cSensorEnvoltura(Address,    PortName,
SpeedInt)
             _wSensor.Propietario = Me
             _wSensor = Resource
          End Sub
```

```
      Public Function ReadSensor(cmd As String) As
Byte()
             Return _wSensor.Read(cmd)
          End Function
```

En esta clase se define un objeto Dispositivo del tipo envoltura (la siguiente etapa de ArquiTAM) al cual se le envían parámetros para su utilización como dirección, puerto y velocidad (necesarios para la utilización del protocolo de comunicación). También se tiene una función para el manejo de la variable leída Read.

Clase envoltura

Luego se tiene la clase de Representante llamada SensorEnvoltura y es análogo a ActuadorEnvoltura en el esquema de ArquiTam para el actuador. La Tabla 4.5 muestra su un extracto de su código.

Tabla 4.5 Extracto de código de la clase envoltura (Sensor)

```
Dim Device As COM_Module

Public Sub New()
        Device.initCom()
    End Sub

Public Function Read(CMD As String) As Val()
        Device.Read(command)
    End Function

Public Sub aWrite(PIN As String, CYCLE As
Byte)
        Device.writeRead(command)
    End Sub
```

En esta clase igualmente hay que iniciar el módulo de comunicación, se establece la forma de leer el valor y de la señal de control como command. Y es suficiente para su función de interfaz de ligue entre el objeto como un elemento general y uno particular.

Clase propulsor

Como última etapa de ArquiTAM se tiene al módulo de comunicación o propulsor, llamado SensorDriver o ActuadorDriver en caso de la clase para el actuador mostrado en la Tabla 4.6.

Tabla 4.6 Extracto de código de la clase propulsor (Sensor)

```
Property State As COM_Module
Property            Devices_reg            As
DeviceInformationCollection

Public Sub initCom ()
        Dim       settings       =       New
ConnectionSettings(SlaveAdd)
    End Sub
```

```
Public Function Read_Data(buffer) As Val()
        Device.Read_Data(buffer)
    End Function

Enum States(PIN As String, CYCLE As Byte)
        Devices_MASTER
        Devices _SLAVE
        Devices _SYSTEM_OK
        Devices _SYSTEM_FAULT
        Devices _SYSTEM_NOINIT
        Devices _SYSTEM_BUSY
    End enum
```

En esta parte del código se inicializa la interfaz de comunicación con las características descritas específicamente con anterioridad (velocidad, puerto, dirección, etc…), se utilizan estados propios para la verificación de la comunicación entre dispositivos y se obtiene el valor leído, el cual es enviado a la envoltura, luego al representante y finalmente al controlador para de ahí utilizar su valor para despliegue de información y del cálculo de la señal de control.

En el caso de las clases para el actuador, como es mencionado a lo largo de esta sección son análogos a los del sensor con sus particularidades propias descritas por el usuario pero igual en su estructura y funcionamiento a estas, solo que en vez de lectura es la escritura de valores. Para el caso de la clase Actuador funciona al igual que la clase propulsor, ligada directamente a la clase Actuador, obtiene el valor del sensor de la clase control y realiza la operación de control directamente relacionada con dicho elemento, y recae en el usuario proporcional o añadir los módulos de control, como así los de comunicación deseados para su implementación en la incubadora.

V. Sistema en Operación

En este capítulo se hace se presenta la operación del sistema partiendo del proceso de configuración inicial del equipo, describiendo la manera de añadir y configurar, los distintos elementos que conforman a un sistema de este tipo (específicamente configurando el controlador de temperatura), donde primero se añade un sensor configurando las variables necesarias para su implementación, luego se añade un actuador; posteriormente se identifica y configura el controlador esto es el tipo de objeto con el modo de control a emplear, asociación de la variable a controlar, además de la configuración del propio controlador , y definición del valor de referencia.

5.1 Inicio de operación

Al iniciar el equipo se muestra la pantalla NavegationPage. Dentro del elemento Frame se encuentra la Page0 como se menciona en el capítulo anterior, sirviendo así como una pantalla de bienvenida (Figura 5.1). De aquí se puede acceder a la página que se desee, sin embargo el primer paso para empezar a utilizar el sistema es el de instanciar o crear un objeto sensor y/ o actuador, con lo que se habilita la creación de un controlador. Además siempre se podrá ver la página de información para mayor detalle.

Figura 5.1 Pantalla de bienvenida

5.2 Configuración de elemento sensor

Como es mencionado en la sección 5.1 lo ideal (pero no obligatorio) siguiendo el flujo natural de la creación de un controlador es iniciar configurando un nuevo objeto sensor dando clic en el botón “New Sens” en la pantalla de bienvenida. Al hacer esto en el Frame de la NavegationPage se muestra la Pagina1 como se puede observar en la Figura 5.2. En esta pantalla el usuario ingresa los valores de cada campo donde a manera de ejemplo en este caso serán LM35 para ID, Temperatura para Nombre, °C para Unidades, Address o Dirección con un valor de 0xE3 (dirección del sensor físico en el Arduino a través de I^2C) y los valores aproximados de los tres parámetros del polinomio característico. Posteriormente se da clic en el botón “Crear”.

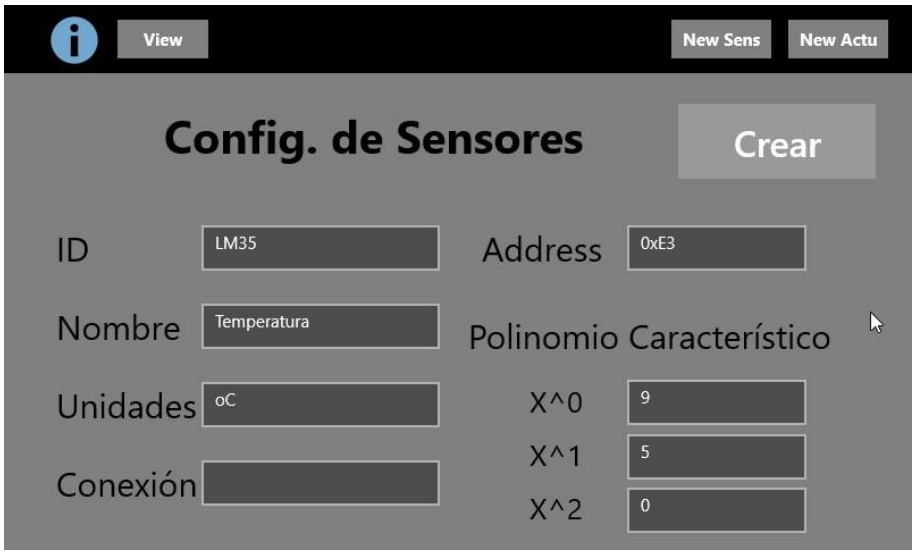


Figura 5.2 Pantalla de configuración de sensores

En este momento el usuario al dar clic en “Crear” dentro del Frame se cargará la pantalla de lectura de valores (la cual puede ser accedida a través de el botón “View” en cualquier momento) para poder empezar con la lectura de los valores de los sensores configurados. Como se puede apreciar en la Figura 5.3 se muestra la pantalla de lectura de valores, donde la línea de texto muestra el Nombre, el valor calculado a partir de la lectura obtenida del sensor y las Unidades dadas por el usuario.

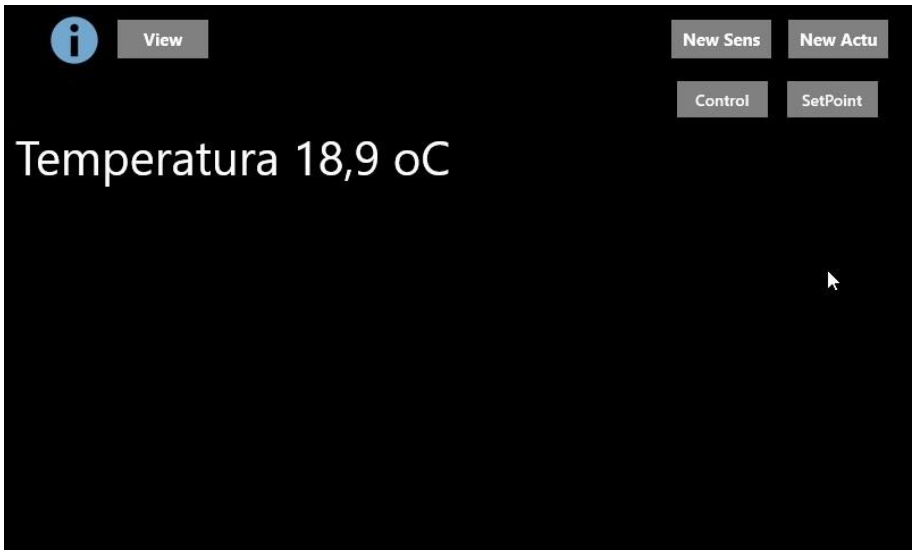


Figura 5.3 Pantalla de lectura de variables

5.3 Configuración de elemento actuador

Ahora, para la creación de un objeto actuador se da clic en el botón “New Actu”, el cual muestra la Page2 (Figura 5.4), donde el usuario asigna su identificador ID y una dirección para el Pin de la salida del actuador, en este caso con los valores de Resistencias y 0xA1 (dirección del actuador físicamente en el Arduino a través de I^2C) respectivamente, con eso será suficiente por ahora.

Figura 5.4 Pantalla de configuración de actuadores

El siguiente paso consiste en crear un objeto controlador (clase que implementa la acción de control) parar poner en acción al actuador. La creación del controlador asocia a ambos objetos sensor y actuador, donde el controlador realizará la acción de control con base a los parámetros establecidos y con respecto a la medición del sensor seleccionado. Para tal efecto, una vez cargada la página mostrada en la Figura 5.5, (dirigiéndose a la página de visualización de mediciones y dando clic en el botón de Control) aparecerán dos listas, a la izquierda la de los sensores creados y a la derecha la de los actuadores; el usuario deberá seleccionar un objeto de cada lista para ligar su funcionamiento en el nuevo objeto controlador que será creado. Ahora el usuario deberá ingresar los parámetros para el funcionamiento deseado del controlador, como lo es un identificador ID, el valor de referencia deseado (Set Point), una ganancia en caso de realizar una acción que lleve una función Proporcional.

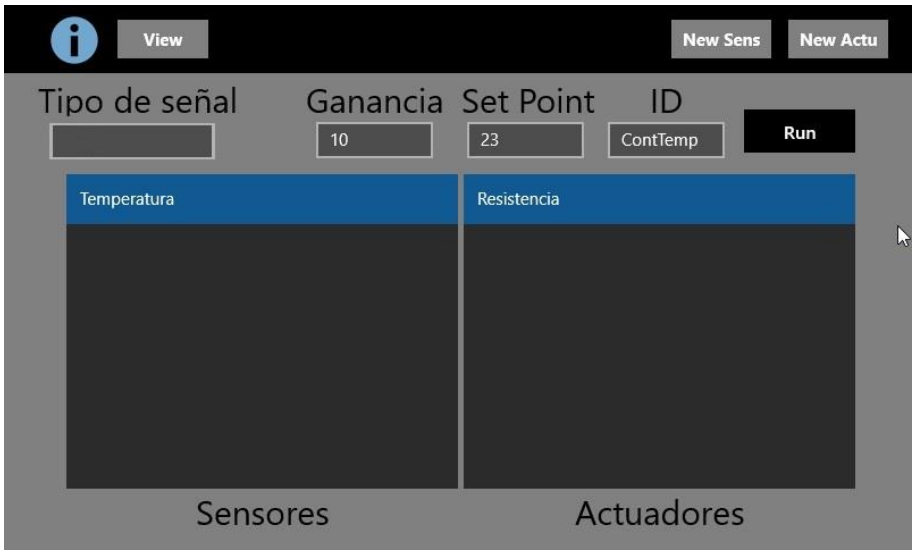


Figura 5.5 Pantalla de configuración de controladores

Habiendo realizado los pasos anteriores se puede decir que la configuración para el control de una variable está completa. El proceso se repite por cada variable a controlar. Ahora bien, en el caso de que se requiera configurar el valor de referencia deseado después de la creación del controlador, el usuario se debe dirigir a la pantalla de configuración de valor de referencia. En esta pantalla el usuario selecciona de una lista el controlador asociado al punto de referencia (Set Point) a establecer el controlador que contenga el valor deseado a configurar.

5.4 Configuración del Punto de Referencia (Set Point)

Para la configuración del valor de referencia (mostrada en la Figura 5.6) hay que dirigirse primero a la pantalla de visualización de datos (clic en “View”), y dar clic en el botón de “SetPoint”. Aquí se mostrara una lista de los controladores creados, serán identificados por el ID dado al momento de su creación, al seleccionarlo en la parte derecha se mostrara a que sensor y actuador pertenece (igualmente por el identificador anteriormente dado), así como el valor del punto de referencia, solo será cuestión de que el usuario ingrese el valor nuevo dentro del cuadro de texto y de clic en guardar para que se modifique dicho valor.

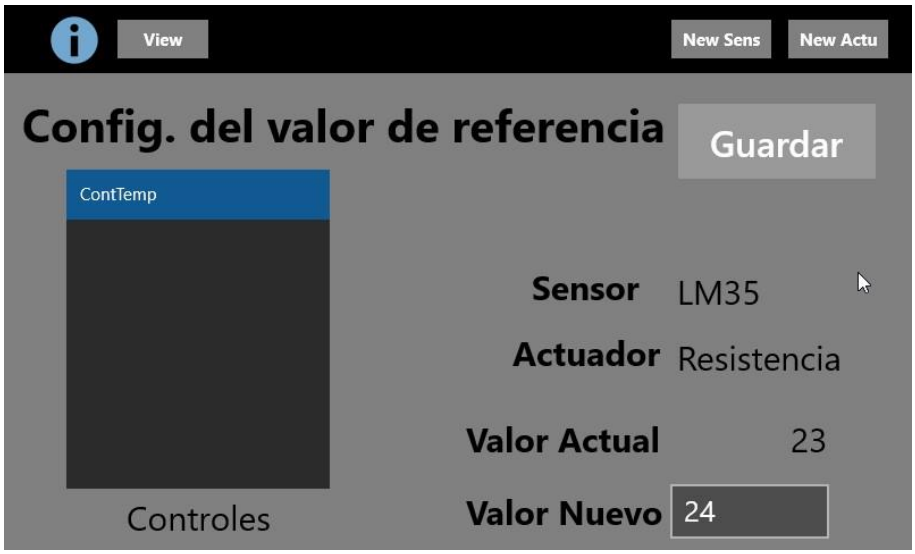


Figura 5.6 Pantalla de configuración del valor de referencia

Como su nombre lo indica, el valor actual muestra el valor presente configurado para el actuador seleccionado dentro de la lista izquierda (Controles), y el campo de valor nuevo permite escribir el valor nuevo a guardad para el controlador. Al igual que en las pantallas de configuración pasadas (configuración de sensor, actuador y controlador) al da clic en "Crear" o en este caso "Guardar" se pasa directamente la pantalla de lectura de variables con los cambios efectuados.

5.5 Configuración de elemento actuador

Siempre es de utilidad contar con información que describa las características y funciones de un sistema, como es el caso de esta incubadora, se cuenta con una pantalla de información para poder recordarle al usuario las diferentes funciones que tiene y definir las variables que le son solicitadas al usuario al momento de configuración de los elementos que conforman al sistema. El icono superior izquierdo ("i") muestra la página de información (Figura 5.7), para salir de esta solo es necesario pulsar cualquiera de los botones a donde se desee ir.

Figura 5.7 Pantalla de información

VI. Conclusiones

6.1 Conclusiones del trabajo de tesis

El sistema de control implementado con base en el mecanismo de acoplamiento débil, es un sistema flexible respecto a la configuración de sensores, actuadores y acción de control a emplear. La idea original, respecto al nivel de flexibilidad, en donde el número y tipo de sensores / actuadores, así como el algoritmo de control, no requieran ser definidos en tiempo de diseño no es alcanzado dado la falta de soporte de Windows 10 IoT para la incorporación dinámica de código. Sin embargo, el sistema se encuentra preparado para un eventual soporte de este tipo de incorporación de código. Debido a tal preparación, en el sistema desarrollado agregar uno de estos elementos es relativamente simple, aún y cuando se requiere abrir el código fuente, la estructura y lógica de operación, permite hacerlo simplemente agregando la clase correspondiente en la estructura de clases del sistema. La incorporación dinámica de código en Windows 10 IoT ha sido discutida por los propios desarrolladores de aplicaciones y Microsoft para poder agregarlo como característica en futuras versiones de la plataforma.

Por lo que en caso de que el usuario desee utilizar un protocolo de comunicación con los sensores / actuadores diferente (no implementado en el sistema) sólo tiene que reemplazar el archivo del driver por el del nuevo deseado. Para esto debe detener la ejecución del programa y cargar los nuevos archivos a la tarjeta. Una vez que Windows IoT permita la carga de código en tiempo de ejecución el sistema está preparado para su uso. El usuario no tendrá que realizar todo este proceso y el acoplamiento débil en el equipo se encontrara en un nivel aun mayor del actual, ya que hay que recordar que si bien no se puede utilizar esta característica, el Framework se encuentra creado y funciona al cambiar los archivos de drivers, que en el caso de que se habilitara esta función sería simplemente cargar los DLL con las funciones deseadas.

Si bien la plataforma universal de Windows brinda una gran cantidad de posibilidades, el estado actual de Windows IoT requiere de bastante desarrollo y seguimiento por parte de Microsoft, desde de dar soporte a dispositivos USB de manera “Plug&Play” como el corregir los problemas de seguridad que pueda tener y evitando de esta manera la carga de código en tiempo de ejecución. Afortunadamente la tendencia de los esfuerzos por parte de Microsoft es la de diseño de entornos para aplicaciones de dispositivos inteligentes con soporte al IoT, lo que asegura un buen futuro para esta plataforma aún más teniendo en cuenta la gran capacidad que tiene al contar con poder hacer uso de la UWP.

Respecto a la aplicación del esquema de referencia ArquiTAM, a pesar de la falta de ciertas funcionalidades del sistema (por limitaciones de Windows IoT), que se espera sean temporales, se pudo lograr satisfactoriamente la implementación del mecanismo de acoplamiento débil, en un ambiente orientado a objetos. Sin embargo, el concepto de

operación dirigido por modelo, y el empleo de la técnica de modelado iMRP, no fue necesaria su aplicación en el presente sistema dado que la secuencia de operación del sistema es relativamente fija, usar una técnica de modelado, para la interpretación del modelo en tiempo de operación, resulta un esfuerzo desproporcionado para su uso en este tipo de sistemas.

6.2 Trabajo futuro

Una vez exista soporte para “Reflection” en Windows 10 IoT agregar dicha característica al sistema para poder cargar la configuración de propulsor a través de archivos DLL y conseguir un mayor dinamismo en la flexibilidad de protocolos de comunicación del sistema.

REFERENCIAS

[1] Baker, J. P. (1996). The Machine In The Nursery: Incubator Technology And The Origins Of Newborn Intensive Care. Baltimore: Johns Hopkins University Press.

[2] Centro Nacional de Excelencia Tecnológica en Salud (CENETEC. (2014). Guía Tecnológica No. 4. Guías Tecnológicas, Incubadora Neonatal (GMDN 36025 y 35121), 47.

[3] Alistair G. S. Philip. (October, 2005). The Evolution of Neonatology. Pediatric Research, 58(4):799-815.

[4] Baker, J. P. (2000). The Incubator and the Medical Discovery of the Premature Infant. Journal Of Perinatology, 20(5), 321.

[5] Oliveira Gomes, Tatiana; Neves Sant'Anna, Andreia; de Vasconcellos Braga, Amanda;Rocha Porto, Fernando (2017). The premature newborn in mid-twentieth century according to Julius Hes. Revista de Pesquisa Cuidado éFundamental - E-ISSN: 2175-5361, vol. 9, núm. 4, pp. 955-961.

[6] Gordo-Vidal, F., & Enciso-Calderón, V. (2012). Síndrome de distrés respiratorio agudo, ventilación mecánica y función ventricular derecha. Medicina Intensiva, 36(2), 138-142. Recuperado el 11 de mayo de 2018, de: http://scielo.isciii.es/scielo.php?script=sci_arttext&pid=S0210-56912012000200007&lng=es&tlng=pt

[7] Dudrick SJ, Wilmore DW, Vars HM, Rhoads JE (1968) Long-term total parenteral nutrition with growth, development, and positive nitrogen balance. Surgery 64:134–142

[8] Silverman, W.A. (1980). A Modern Parable. Retrolental Fibroplasia Grune and Stratton, New York.

[9] Arroba Ingeniería. (2018). saps ISOTERM.). Recuperado el 12 de mayo de 2018, de: http://www.arrobaing.com.mx/ incubadoras.html

[10] DRAGER. (2018). Dräger Babytherm 8004/8010. Recuperado el 12 de mayo de 2018, de: https://www.draeger.com/esmx/Hospital/Products/Thermoregulation-and-Jaundice-Management/Neonatal-Open-Care/Babytherm-8000-8010

[11] GE Healthcare. (2018). Giraffe Incubator Carestation. Recuperado el 13 de mayo de 2018, de: http://www3.gehealthcare.com/en/products/categories/maternal-infant_care/giraffe_incubator_carestation

[12] Air-Shields. (2018). Air-Shields Incubadora para neonatos Isolette C2000. Recuperado el 13 de mayo de 2018, de: http//www.ingenieriabahia.com.ar/manuales/c2000.pdf

[13] FANEM. (2018). FANEM Multisystem 2051. Recuperado el 14 de mayo de 2018, de: http://www.hospi-medics.com/neonatal/PDF/multisytem 202051.pdf

[14] S. d. s. d. g. d. l. E. U. Mexicanos, Norma oficial mexicana NOM-066-SSA1-1993, que establece las especificaciones sanitarias de las incubadoras para recién nacidos, México, D.F.: Normas oficiales mexicanas, 1994.

[15] Acosta Cano de los Ríos, José Eduardo (2015). Esquema de Referencia para Acoplamiento Débil entre Sistema Informático y Equipo de Producción. Tesis (Doctoral), E.T.S.I. Industriales (UPM)

[16] Acosta, J.A., & Sastrón, F. (2006). Schematic Architecture: Reference Architecture / Frameworks / Particular Models for the Shop Floor Environment. IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics, 4563-4568.

[17] Thompson, James. (2003). Organizations in Action: Social Science Bases of Administrative Theory.

[18] Talcott Parsons (1960). Structure and Process in Modern Societies. Glencoe, Illinois: The Free Press. 344 pp., Social Forces, Volume 39, Issue 1, October 1960, 84–85.

[19] Orton, J. D., & Weick, K. E. (1990). Loosely coupled systems: A reconceptualization. Academy of Management Review: ABI/INFORM Global. 203.

[20] Acosta Cano de los Ríos José Eduardo, Franco Luna Fabián, Óscar Chávez López. (Oct 2019). Desarrollo de un Controlador Débilmente Acoplado para Incubadoras Neonatales Implementado en UWP. Congreso Internacional en Ing. Electrónica. Mem. ELECTRO, Vol. 41, Pag. 1-6.

[21] Margaret Rouse. (2008). The vital guide to modern programming languages and their uses. 2019, de TechTarget - Search App Architecture Sitio web: https://searchapparchitecture.techtarget.com/definition/object-oriented-programming-OOP

[22] Microsoft Support. (2019). What is a DLL?, de Microsoft Sitio web: https://support.microsoft.com/en-us/help/815065/what-is-a-dll

[23] Windows Dev Center. (2018). Suggested Prototype Boards. 2019, de Microsoft Sitio web: https://docs.microsoft.com/en-us/windows/iot-core/tutorials/quickstarter/PrototypeBoards

[24] Zaragoza, Y. Gómez, A. Cabrera, G. Trujano, C. Campos, S. Montoya, A. García, S. Jiménez, I. Chairez, B. Aguilar, I. Valencia, K. Mendoza, C. Rodríguez, E. Bautista, G. Gálvez. (2001). DISEÑO Y CONSTRUCCIÓN DE UN PROTOTIPO DE INCUBADORA CONTROLADO POR LÓGICA DIFUSA. Instituto Politécnico Nacional. Unidad Profesional Interdisciplinaria de Biotecnología

[25] Restrepo Pérez, Laura, Durango Londoño, Natalia, Gómez Suárez, Nicolás, González Ramírez, Felipe, & Rivera Bonilla, Nadia. (2007). PROTOTIPO DE INCUBADORA NEONATAL. Revista Ingeniería Biomédica, 1(1), 55-59. Recuperado el 10 de mayo, 2018, de: http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1909-97622007000100012&lng=en&tlng=es

[26] Gabriel Vinicio Moreano Sanchez. (2015). DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE CONTROL DE TEMPERATURA Y HUMEDAD PARA UN PROTOTIPO DE INCUBADORA NEONATAL QUE INCLUYA MONITOREO DE SIGNOS VITALES. Escuela Politécnica Nacional, República del ecuador.

[27] Simons, R. (1994). How new top managers use control systems as levers of strategic renewal. Strategic Management Journal, 15(3), 169–189.

[28] Anthony, R. N. & Govindarajan, V. (2001). Management control systems / Robert N. Anthony, Vijay Govindarajan. 10th ed. Boston: McGraw Hill/Irwin.

[29] Wall, Friederike. (2019). Adaptation of the boundary system in growing firms: an agent-based computational study on the role of complexity and search strategy. Economic Research-Ekonomska Istraživanja. 1-23.

[30] J. Acosta-Cano, F. Sastrón-Báguena. (June 2013). Loose Coupling Based Reference Scheme for Shop Floor-Control System/Production-Equipment Integration. Journal of Applied Research and Technology, 11, 447-469.

[31] Sistema de Manufactura Reconfigurable y Competitividad Industrial Vol. 1 #2, 2010. pp. 97-113

[32] INEGI (2015). Encuesta Intercensal. Recuperado el 17 de marzo de 2018, de: http://www.beta.inegi.org.mx/proyectos/enchogares/especiales/intercensal/

[33] H. Chandra, E. Anggadjaja, P. S. Wijaya and E. Gunawan, "Internet of Things: Over-the-Air (OTA) firmware update in Lightweight mesh network protocol for smart urban development," 2016 22nd Asia-Pacific Conference on Communications (APCC), Yogyakarta, 2016, pp. 115-118.

[34] Xavier Ferré Grau, María Isabel Sánchez Segura. (2011). Desarrollo Orientado a Objetos con UML. 2019, de Facultad de Informática – UPM

[35] G. Booch, I. Jacobson, J. Rumbaugh. (1999). El Lenguaje Unificado de Modelado. X: Addison Wesley.

[36] Texas Instruments. (1999). LM35 Precision Centigrade Temperature Sensors. 2017, de TI Sitio web: http://www.ti.com/lit/ds/symlink/lm35.pdf

[37] YSI. (2013). YSI Precision Thermistors & Probes. 2018, de Advanced Industrial Systems, Inc. Sitio web: http://www.advindsys.com/Manuals/YSIManuals/YSICatalog-1998.pdf

[38] PROCESS SENSING TECHNOLOGIES. (2018). Replacement galvanic oxygen sensor for use in Biomed Device Cross Vent w/ color display (PSR-11-917-MHJ). 2018, de Analytical Industries Inc. Sitio web: https://aii1.com/oem_replacement/PSR-11-917-MHJ.htm

[39] Seeed. (2015). Grove - Gas Sensor (O2). 2018, de SeeedStudio Sitio web: https://www.mouser.com/ds/2/744/Seeed_101020002-786509.pdf

[40] PLUX. (2015). SpO2 Sensor. 2018, de BioSignalsPlux Sitio web: https://biosignalsplux.com/datasheets/SpO2_Sensor_Datasheet.pdf

[41] Honeywell. (2010). HIH-4000 Series Humidity Sensor. 2018, de Honeywell International Inc. Sitio web: https://sensing.honeywell.com/honeywell-sensing-hih4000-series-product-sheet-009017-5-en.pdf

[42] SENSIRION. (2008). Datasheet SHT1x (SHT10, SHT11, SHT15) Humidity and Temperature Sensor. 2018, de SENSIRION Sitio web: https://www.sparkfun.com/datasheets/Sensors/SHT1x_datasheet.pdf

[43] OSEPP. (2013). DHT 11 Humidity & Temperature Sensor. 2017, de Mouser Electronics Sitio web: https://www.mouser.com/ds/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

[44] SUNFOUNDER. (2017). Analog Temperature Sensor Module. 2018, de SUNFOUNDER Sitio web: http://wiki.sunfounder.cc/index.php?title=Analog_Temperature_Sensor_Module

[45] Harrington, W. (2015). Learning Raspbian. Packt Publishing Ltd.

[46] Borycki, D. (2017). Programming for the Internet of Things: Using Windows 10 IoT Core and Azure IoT Suite. Microsoft Press.

[47] Windows Dev Center. (2019). Windows 10 SDK. 2019, de Microsoft Sitio web: https://developer.microsoft.com/en-us/windows/downloads/windows-10-sdk