# DISEÑO DE UNA PLATAFORMA SOFTWARE QUE APOYE LOS PROCESOS DE CRIANZA EN EL ÁREA DE LA PISCICULTURA

FABIO ANDRES MONTAÑEZ GOMEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2017

# DISEÑO DE UNA PLATAFORMA SOFTWARE QUE APOYE LOS PROCESOS DE CRIANZA EN EL ÁREA DE LA PISCICULTURA

FABIO ANDRES MONTAÑEZ GOMEZ

Trabajo de grado para optar el título de Ingeniero de Sistemas

Director
GABRIEL RODRIGO PEDRAZA FERREIRA
PhD. Ciencias de la Computación
Codirector
CARLOS ANÍBAL VÁSQUEZ CARDOZO
Médico Veterinario y Zootecnista, Especialista en Docencia Universitaria y Maestro en Sistemas de Producción Animal

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2017

## DEDICATORIA

Dedico este trabajo de grado a mi madre Carmen Rosa Gomez Amado y a mi padre José Raymundo Montañez Ortega ya que sin el amor que siempre me brindaron, su apoyo, su ejemplo de superación ante las adversidades y estar siempre incondicionalmente, no hubiera podido alcanzar este logro académico que hoy me enorgullece.

A mi hermana Heidy Johanna Montañez Gomez por siempre confiar en mis capacidades, por escucharme y por ser ese gran apoyo moral y darme consejos cuando más los necesitaba. Por ser esa gran mujer que me enseño tantas cosas cuando solo era un niño.

A mi sobrina Kimberly Dayanna Castillo Montañez, que este meta que hoy he logrado, sea una gran motivación para que sigas persiguiendo tus sueños y nunca dejes de hacerlo, que sea una muestra de que todo se consigue con esfuerzo y perseverancia. Adelante kim, ni un paso atrás.

A mi cuñado Luis Alberto Castillo Pico, quien es uno de mis grandes ejemplos a seguir, quien en su momento fue la persona que me inspiro a ser grande en la vida y me ha demostrado de mil maneras que desde que uno haga las cosas bien y sin hacerle daño a nadie, todo va llegando en su momento.

Fabio Andres Montañez Gomez

## AGRADECIMIENTOS

A Elberto Carrillo por haber sido mi primer profesor en este largo camino en el cual decidí formarme y estudiar Ingeniería de Sistemas y haber hecho que me enamorara de la programación. A mi director de proyecto de grado Gabriel Pedraza que gracias a su experiencia y especialmente a su paciencia decidió ser mi mentor en este último paso de mi vida académica en el pregrado. También a todos aquellos profesores con los que en algún momento compartimos el aula de clases y dejaron una parte de su trabajo en mi desarrollo como profesional.

A mis compañeros de la universidad con los que compartimos alegrías y tristezas, Miguel Márquez, Lizeth Rodríguez, David Boada, Marcela León, Wanda Rincón. A mis compañeros Antonio Cortes, Gina Mejía, Daniel López, Diego Ferrin, Harold Ardila, con los cuales dedicamos grandes cantidades de horas de estudio y hoy estamos viendo los frutos de todos esos esfuerzos. A mis compañeros del CEIS especialmente a Jeremías Pabón, Patricia Patiño y Andres Ferreira con los cuales tuvimos la dicha de cambiar radicalmente nuestro centro de estudios. Y a todos aquellos a los que tuve la dicha de conocer y compartir este proceso de vivir académicamente en la Universidad industrial de Santander.

A mis amigos José Serrano, Julián Pérez, Iván Vecino y Andres Castro, los cuales han aportado a mi desarrollo personal y que me enseñaron que lo importante es la calidad de persona que eres, sin su amistad sincera este proceso JAMÁS hubiera sido tan placentero, porque serán mis hermanos para siempre.

A Dios por estar siempre en esos momentos de angustia y desesperación, mostrándome la luz de la esperanza cuando todo era tan oscuro.

Infinitas Gracias

**CONTENIDO**

## LISTA DE FIGURAS

## LISTA DE TABLAS

## GLOSARIO

**Acui:** Significa 'agua'. Acuicultura.

**Acuicultura:** Cultivo de especies acuáticas vegetales y animales. Conjunto de técnicas y conocimientos relativos al cultivo de especies acuáticas.

**Acuicultor (ra):** Perteneciente o relativo a la acuicultura. Persona que se dedica al cultivo de especies acuáticas.

**Piscicultura:** Cría artificial de peces y mariscos. Conjunto de técnicas y conocimientos relativos a la cría artificial de peces y mariscos.

**Piscicultor (ra):** Persona que se dedica a la cría artificial de peces y mariscos.

**Piscícola:** Perteneciente o relativo a la piscicultura. Perteneciente o relativo a los peces.

## RESUMEN

**TITULO:** DISEÑO DE UNA PLATAFORMA SOFTWARE QUE APOYE LOS PROCESOS DE CRIANZA EN EL ÁREA DE LA PISCICULTURA[*]

**AUTOR:** FABIO ANDRES MONTAÑEZ GOMEZ[**]

**PALABRAS CLAVE:** PISCICULTURA, DISEÑO, PLATAFORMA, SENSOR, REGLA, ACTUADOR, SOFTWARE, EMBEBIDO

**DESCRIPCIÓN:**

La toma de decisiones basada en información real y actualizada frente a las distintas problemáticas y necesidades que se presentan en la piscicultura, se ha vuelto un tema bastante relevante, debido al constante cambio en las condiciones ambientales en las que se lleva acabo las actividades piscícolas en Colombia.

Teniendo en cuenta las ventajas del “Internet of Things”, los sistemas embebidos, los sensores y actuadores multipropósito, la portabilidad y la masividad de los frameworks de código abierto, se diseñó una plataforma software que provee al usuario un apoyo para el control del proceso de crianza en la piscicultura; el enfoque del diseño planteado, se basa en la captura de las características del ambiente, para luego mediante un conjunto de reglas analizar en tiempo real los cambios que alteran los ciclos de producción piscícola ejecutando una serie de acciones mediante unos actuadores en pro de corregir dichas alteraciones ambientales.

Al final se evaluó el diseño mediante la implementación de un prototipo funcional que permitió destacar la extensibilidad de la plataforma gracias a la fácil adaptación de nuevos sensores y/o actuadores; La plataforma también fue pensada para ser abierta dando la posibilidad de adaptarse a nuevas necesidades futuras permitiendo que el diseño presentado no se vuelva obsoleto y sea fácil de cambiar.


[*] Trabajo de grado. Modalidad: Trabajo de Investigación.

[**] Facultad de Ingenierías Fisicomecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Gabriel Rodrigo Pedraza Ferreira. Co-Director: Carlos Aníbal Vásquez

## ABSTRACT

**TITLE:** DESIGN OF A SOFTWARE PLATFORM TO SUPPORT AGING PROCESSES IN THE PISCICULTURE AREA[*]

**AUTHOR:** FABIO ANDRES MONTAÑEZ GOMEZ[**]

**KEYWORDS:** PISCICULTURE, DESING, PLATAFORM, SENSOR, RULER, ACTUATOR, SOFTWARE, EMBEDDED.

**DESCRIPTION:**

Making decisions based in real and update information in front to the different problems and needs presents in the pisciculture, it has become a relevant issue. Due to constant change in the environmental conditions in which it is carried out fish farming activities in Colombia.

Taking into account the advantages of “Internet of Things”, the embedded system, sensors and multipurpose actuators, portability and the massiveness of open source frameworks, A software platform was designed that provides to the user a support to the control of the breeding process in the pisciculture; the approach of design, is based on the capture of the characteristics of the environment, then by means of a set of rules analyze in real time the changes that alter the cycles of fish farming production executing a series of actions by means of actuators in order to correct these environmental alterations.

At the end, the design was evaluated through the implementation of a functional prototype that allowed to emphasize the extensibility of the platform thanks to the easy adaptation of new sensors and / or actuators; The platform was also thinking to be open giving the possibility of adapting to the new future needs, allowing that the design presented doesn't become obsolete and be easy to change.


[*] Bachelor Thesis. Modality: Research work

[**] Faculty of Physical-Mechanical Engineering. School of Systems Engineering and Informatics. Advisor: Gabriel Rodrigo Pedraza Ferreira. Coadvisor: Carlos Aníbal Vázquez.

# INTRODUCCIÓN

En pleno siglo XXI la humanidad ha tenido el gran deseo por sistematizar todo lo que está a su alcance y si volteamos a mirar las grandes sociedades podemos afirmar con certeza que un buen ejemplo de este gran deseo son las “Smart Cities” se podría decir que han tenido grandes avances en materia de evolución tecnológica y que el objetivo en este sector va por buen camino; otro sector como la agricultura industrial también está alcanzando el objetivo debido a la fuerte incursión de la industria en el sector agrícola, pero que pasa si miramos la agricultura tradicional, ha tenido pequeños pasos en la adaptación de herramientas tecnológicas, pero, su desarrollo está muy por debajo de otros sectores, como los mencionados anteriormente.

En el siguiente proyecto de grado se presenta el diseño de una alternativa software basada en sensores y actuadores para apoyar al crecimiento y fortalecimiento de la piscicultura en la captación y recolección de los datos que un ambiente piscícola pueda suministrar y de esta manera poder procesarlos y transformarlos en información que permita poder tomar decisiones más acertadas en tiempo real que beneficien de manera positiva los ciclos de producción y la crianza, todo esto desde el punto de vista de la Ingeniería de Sistemas y aún más importante utilizando tecnologías de bajo costo comparadas con las que se utilizan en el sector industrial.

Paralelamente también se desarrolló un prototipo el cual se sometió a un conjunto de pruebas en un escenario real, para poder evaluar el alcance y la eficiencia del diseño que se planteó y de esta manera conocer el trabajo a futuro y las conclusiones finales de este trabajo de grado.

# 1. PLANTEAMIENTO DEL PROBLEMA

La acuicultura es el cultivo de especies y organismos acuáticos vegetales y animales. La piscicultura (término más utilizado en Colombia) por su parte, sólo contempla el conjunto de técnicas y conocimientos relativos a la cría artificial de peces y mariscos. Esta actividad supone la intervención humana en el proceso de producción, cultivo, protección y comercialización de las especies acuáticas involucradas. La producción mundial de pescado sigue creciendo a un ritmo más rápido que la población mundial y la acuicultura se mantiene como uno de los sectores de producción de alimentos de más rápido crecimiento. En 2012, la acuicultura estableció otro máximo histórico de producción y ahora proporciona casi la mitad del pescado destinado a la alimentación humana. [1]

*"La tecnología y los sistemas utilizados en la acuicultura han progresado aceleradamente en los últimos años. Varían desde unos muy sencillos, (como los estanques familiares) hasta otros de alta tecnología (como los sistemas cerrados de producción intensiva para exportación). Gran parte de la tecnología que se utiliza en la acuicultura es relativamente sencilla, muchas veces basada en pequeñas modificaciones que aumentan las tasas de crecimiento y supervivencia de las especies, es decir, mejoran los alimentos, las crías recién nacidas (alevines), los niveles de oxígeno, la protección frente a los depredadores, etc. Casi la mitad de la producción acuícola mundial consta de sistemas sencillos de pequeños estanques de agua dulce, destinados a la cría de peces"*[1]. [2]

En Colombia la piscicultura de agua dulce y marina comenzó en los años ochenta y desde entonces ambos sectores han crecido de manera significativa debido a

[1] Aquaculture topics and activities. Tecnología de la acuicultura. In: Departamento de Pesca y Acuicultura de la FAO [en línea]. Roma. Actualizado 2006 15 09. [Citado 19 October 2016]. http://www.fao.org/fishery/technology/aquaculture/es

que el país tiene varias zonas de temperatura relativamente constantes, climas y microclimas durante todo el año, además, el cultivo de peces de agua dulce se lleva a cabo en todo el país (Ver Figura 1), pero especialmente en la región andina. [3]

*Figura 1. Principales zonas de producción en Colombia por departamentos*


FUENTE: Datos sub-nacionales e individuales a nivel de granjas en Colombia según la Organización de las Naciones Unidas para la Alimentación y la Agricultura. [4]

Los principales mercados a los que exporta Colombia son Estados Unidos, Japón, algunos países de la UE (Italia, España, Francia, Alemania, Bélgica, Reino Unido y Portugal), y en algunos países de América Latina (México, Panamá, Costa Rica, El

Salvador y Puerto Rico). Las exportaciones incluyen el atún, langosta, el pargo, la trucha arco iris, entre otras especies. En los últimos años, debido a los fenómenos climáticos (Calentamiento Global, fenómeno del niño) y la marcada variabilidad de las poblaciones de peces, la pesca ya no es una actividad permanente para la mayoría de los pescadores. Esta situación repercute sobre la economía del país y es donde la piscicultura se impone como una solución para suplir la deficiencia del mercado. [5]

No obstante, se debe tener en cuenta que actualmente la práctica de la piscicultura no industrializada se desarrolla en gran parte por las poblaciones campesinas, por ende, tenemos una falta de conocimiento profesional en el área, lo que conlleva a una técnica más tradicional (artesanal), desconociendo las bondades de la ingeniería en este campo y como resultado unos pequeños márgenes de diferencia respecto a los costes de producción. Un mejor conocimiento de las interacciones entre las bacterias, los nutrientes y los organismos de un ambiente en donde se lleve a cabo el proceso piscícola, han permitido la creación de sistemas cerrados, que tienen la ventaja de reducir al mínimo el riesgo de contraer enfermedades, disminución de efectos secundarios en las especies, ciclos de crianza más óptimos, toma de decisiones más acertadas, entre otros beneficios que los sistemas acuáticos naturales no pueden controlar.

# 2. OBJETIVOS

## 2.1 OBJETIVO GENERAL

Diseñar una plataforma software que provea al usuario un apoyo para el control del proceso de crianza en el área de la piscicultura en un ambiente cerrado.

## 2.2 OBJETIVOS ESPECÍFICOS

- Identificar las necesidades que se presentan en el proceso de crianza de peces en el contexto de la piscicultura en Colombia.

- Diseñar una plataforma software escalable y abierta que tenga como propósito aportar soluciones a las necesidades identificadas.

- Desarrollar un prototipo de la plataforma que utilice hardware disponible en el mercado.

- Validar el prototipo a través de un marco de simulación y/o pruebas de campo.

# 3. ESTADO DEL ARTE

Es importante tener en cuenta aquellos aportes que otros colegas han hecho al tema y que de una u otra manera están relacionados con el presente trabajo de grado, por lo tanto, se hará referencia a algunos de esos proyectos.

En cuanto a piscicultura:
Diseño mecánico a bajo costo y construcción del modelo de una planta de tratamiento de agua para piscicultura ornamental. Este trabajo de grado tenía como objetivo plantear el diseño de una planta de recuperación de aguas para su reutilización en actividades piscícolas, aportando una solución desde el punto de vista mecánico utilizando procesos económicos y seguros de potabilización del agua, asegurando un ambiente saludable para especies ornamentales. [6]

Estudio de caso sobre la utilización de sistemas de generación fotovoltaicos en cultivos de piscicultura intensiva en el departamento de Santander. Este trabajo de grado plantea la factibilidad de implementar un sistema de abastecimiento de energía eléctrica basado en sistemas y tecnologías de generación fotovoltaica garantizando un suministro parcial o total de la demanda de energía en los procesos de producción piscícola intensiva. [7]

En cuanto a sistemas embebidos:
Diseño de un sistema embebido que permite hacer seguimiento y control a vehículos de transporte publico basado en BRT. Este trabajo de grado propone el diseño de una arquitectura que se encarga de recolectar datos por medio de sensores instalados a un sistema embebido en los buses que hacen parte del sistema BRT para luego transmitirlos a una plataforma back-end. [8]

# 4. MARCO TEÓRICO

Para darle una solución viable al problema planteado en el capítulo 2, se pensó abordar el proyecto desde las siguientes temáticas:

## 4.1 PISCICULTURA EN COLOMBIA

“En general Colombia tiene la característica de ser un país tropical, también posee todos los pisos térmicos, la mayoría del año con temperaturas estables y una red fluvial que cubre todo el país. En cuanto a recursos hídricos cuenta con una gran cantidad de cuencas hidrográficas y tiene una amplia superficie continental (1´441.748 km2), también cuenta con 2 costas, una sobre el océano Atlántico y la otra en el océano Pacífico. Hablando sobre acuicultura, cuenta con terrenos aptos que le otorgan un potencial para su desarrollo y una de las mayores diversidades de peces de todo el planeta, al igual que aguas dulces, marinas, salobres y una alta biodiversidad de organismos hidrobiológicos. A diferencia de las actividades agropecuarias tradicionales, la acuicultura ha tenido un buen ritmo de crecimiento generando una marcada rentabilidad. Además, está contribuyendo a sustituir parte de la disminución de la oferta natural del recurso pesquero, debido a factores como degradación del hábitat, sobrepesca, factores ambientales, etc. Cuenta con entes estatales y privados que generan políticas para su desarrollo, que la apoyan y la promueven, mediante programas de administración, ordenamiento, investigación, etc.

La acuicultura de agua dulce tiene sus inicios en los años 30 con la introducción de la trucha arco iris (Oncorhynchus mykiss) con fines para la pesca deportiva. Más adelante en los 70´s se realizan investigaciones sobre la biología y el cultivo de la ostra de mangle (Crassostraea rhizhophorae) en aguas salobres y marinas.

En los años 80 inicia el desarrollo de la camaronicultura y el cultivo de camarón patiblanco (Penaeus vannamei) en el Atlántico y en el Pacífico colombiano con fines de exportación, a través del apoyo de la misión China al país por intermedio del INDERENA (INPA, 1995). en los años 90 el desarrollo de la tecnología de cultivo de la tilapia roja (Oreochromis sp) y a inicios del 2000, se comienzan las investigaciones con la adaptación al cautiverio y la reproducción del pargo lunarejo (Lutjanus guttatus) en el Pacífico y con la reproducción del Pargo palmero (Lutjanus analis) y la reproducción y el cultivo de los pectínidos (Argopecten sp y Nodipecten sp) en el Caribe. Cabe resaltar, que se presenta un crecimiento significativo desde 1985 hasta la fecha, demostrando ser una actividad económica altamente contribuyente hacia el futuro de la seguridad alimentaria debido en parte a sus especies nativas como lo son: el bocachico (Prochilodus magdalenae), el bagre (Pseudoplatistoma fasciatum), las cachamas blanca y negra (Piaractus brachypomus y Colossoma macropomum) y el yamú (Brycon siebenthalae).

Los sistemas de producción acuícolas pueden ser básicamente:

- Estanques en tierra, para especies como la cachama blanca y el camarón patiblanco.
- En clima cálido se utilizan jaulas flotantes para las especies como la tilapia roja.
- En clima frío se emplean estanques en tierra, recubiertos con geomembrana o construidos en cemento para especies como la trucha.

Los técnicos y profesionales dedicados a la acuicultura han evolucionado e incrementado los conocimientos y experiencias en el interior del país, pasando a ocupar cargos de responsabilidad a nivel nacional en centros de investigación, Universidades, Corporaciones Autónomas Regionales, Institutos Nacionales, Ministerios e incluso en sus propias empresas en Colombia o en el exterior."[2]

[2] National Aquaculture Sector Overview. Visión general del sector acuícola nacional - Colombia. National Aquaculture Sector Overview Fact Sheets. Texto de Salazar Ariza, G. In: Departamento de Pesca y

## 4.2 INTERNET DE LAS COSAS (IoT)

El “Internet de las Cosas” (The Internet of Things), consiste en que tanto personas como objetos puedan conectarse a Internet en cualquier momento y lugar. IoT hace referencia a un mundo conectado hasta el último extremo, donde objetos y seres físicos interaccionan con entornos virtuales de datos en el mismo espacio y tiempo. Soñamos con poder medir y controlar por completo nuestro entorno. Esto será posible usando la información extraída a través de millones de sensores que poblarán cada rincón de nuestro entorno y que podrán estar integrados en cualquier objeto de nuestra vida cotidiana. [9] Las posibles aplicaciones de IoT son numerosas y diversas, tocando prácticamente todos los ámbitos de la vida cotidiana, de las empresas y la sociedad en su conjunto. A continuación, se muestran aplicaciones específicas en la agricultura.

- Mejora de la calidad del vino:

Monitoreo de humedad del suelo y diámetro del tronco en las cepas para controlar la cantidad de azúcar en las uvas y la salud de la vid.

- Casas Verdes:

Control de las condiciones micro-climáticas para maximizar la producción de frutas y hortalizas y su calidad.

- Campos de golf:

Riego en las zonas secas para reducir el agua necesaria para su mantenimiento.

- Red de estaciones meteorológicas:

Estudio de las condiciones climáticas en los campos para pronosticar la formación de hielo, la aparición de lluvia, sequía, nieve o el cambio del viento.

- Compost:

El control de los niveles de humedad y temperatura en la alfalfa, heno, paja, etc.

- Cuidado de crías:

---

Acuicultura de la FAO [en línea]. Roma. Actualizado 1 February 2005. [Citado 24 de October 2016]. http://www.fao.org/fishery/countrysector/naso_colombia/es#tcN7015F

Control de las condiciones de crecimiento de las crías de los animales para garantizar su supervivencia y su salud.

- Seguimiento de animales:

Ubicación e identificación de los animales que pastan abiertamente o en establos.

- Control de niveles de gases tóxicos:

Estudio de la ventilación y la calidad del aire en las granjas y detección de gases nocivos proveniente de excrementos. [10]

Respecto a lo que se puede encontrar en el campo de la piscicultura alrededor del mundo podemos encontrar las siguientes aplicaciones:

- Alimentadores automáticos:

Mide por medio de sensores el apetito de los peces y distribuye el alimento en las proporciones adecuadas para evitar pérdidas por la alimentación manual. [11]

- Características del agua:

Mediante sensores que miden profundidad, salinidad y temperatura controlan las condiciones ideales para mantener ciertas especies como las ostras que no pueden vivir en todas las condiciones acuáticas, permitiendo de esta manera, poder cultivar esta especie en regiones que antes no eran adecuadas para su cultivo. [12]

- Información y seguridad:

En mar abierto es difícil controlar y vigilar las bateas (jaulas suspendidas en el mar), especialmente de mariscos, pero una plataforma de boyas conectadas por m2m permite obtener información constante sobre el crecimiento de estas especies, a su vez registrar accesos no autorizados y alertar a las autoridades para disminuir el hurto. [13]

- Piscigranjas “inteligentes”:

En Corea se han instalado sensores en estanques de anguilas midiendo niveles de oxígeno, calidad del agua y temperatura, si se detectan cambios ambientales

que puedan ser fatales para las especies el sistema envía alertas a los dispositivos sincronizados en tiempo real. [14]

## 5. METODOLOGÍA

Se presenta el cronograma que describe la metodología que se llevó a cabo en la ejecución de las fases del proyecto:

*Tabla 1. Cronograma de la metodología de trabajo*

| | SEMANAS | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ACTIVIDADES** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| | **FASE 1: IDENTIFICACIÓN** | | | | | | | | | | | | | | | |
| 1.1 Indagar y documentarse en general en el área de la piscicultura | O | O | O | | | | | | | | | | | | | |
| 1.2 Visualizar el panorama de la piscicultura en Colombia | O | O | O | | | | | | | | | | | | | |
| 1.3 Identificar las necesidades presentes en el proceso de crianza | | O | O | | | | | | | | | | | | | |
| | **FASE 2: ARQUITECTURA** | | | | | | | | | | | | | | | |
| 2.1 Investigar el hardware (sensores y actuadores) que se encuentra disponible en el mercado y aporte una solución al proyecto | | | O | O | | | | | | | | | | | | |
| 2.2 Proponer distintas soluciones de arquitectura | | | | O | | | | | | | | | | | | |
| 2.3 Escoger la arquitectura software | | | | O | O | | | | | | O | | | | | |
| | **FASE 3: DESARROLLO** | | | | | | | | | | | | | | | |
| 3.1 Diseñar un primer prototipo base | | | | | | O | O | | | | | | | | | |
| 3.2 Construir el prototipo | | | | | | | O | O | | | | | | | | |
| 3.3 Construir un segundo prototipo “evolucionado” teniendo en cuenta las observaciones del primer prototipo base | | | | | | | | | | | | O | O | | | |
| 3.4 Codificar el prototipo evolucionado | | | | | | | | | | | | | O | O | | |
| | **FASE 4: VALIDACIÓN** | | | | | | | | | | | | | | | |
| 4.1 Validar el prototipo base | | | | | | | | O | O | | | | | | | |
| 4.2 Realizar un documento con las observaciones, correcciones y sugerencias que arroje la validación del prototipo base | | | | | | | | | O | O | | | | | | |
| 4.3 Validar el prototipo evolucionado mediante pruebas controladas | | | | | | | | | | | | | | O | O | |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 4.4 Documentar el prototipo evolucionado | | | | | | | | | | | | | | | O | O |

El proyecto se ejecutó bajo la metodología de prototipado evolutivo y se realizó en 4 fases, las cuales fueron:

### 5.1 Fase 1: Identificación

Actividades:

1.1 Indagación y documentación en el área de la piscicultura
1.2 Visualización del panorama de la piscicultura en Colombia
1.3 Identificación de las necesidades presentes en el proceso de crianza

Producto:

Documento que contiene las necesidades que se presentan en la piscicultura y específicamente en el proceso de crianza de peces

### 5.2 Fase 2: Arquitectura

Actividades:

2.1 Investigación del hardware (sensores y actuadores) que se encontraba disponible en el mercado a la fecha y aporto solución al proyecto
2.2 Proposición de distintas soluciones de arquitectura
2.3 Selección de la arquitectura software (primer y segundo ciclo)

Producto:

Documento del diseño de la arquitectura utilizada en la ejecución del proyecto

### 5.3 Fase 3: Desarrollo

Actividades:

Primer ciclo:

3.1 Diseño del prototipo base

3.2 Construcción del prototipo

Segundo ciclo:

3.3 Construcción del segundo prototipo “evolucionado” teniendo en cuenta las observaciones del prototipo base

3.4 Codificación del prototipo evolucionado

Producto:

Documento de la codificación final del prototipo evolucionado

### 5.4 Fase 4: Validación

Actividades:

Primer ciclo:

4.1 Validación del prototipo base

4.2 Documento de observaciones, correcciones y sugerencias que arrojo la validación del prototipo base

Segundo ciclo:

4.3 Validación del prototipo evolucionado mediante pruebas controladas

4.4 Documento del prototipo evolucionado

Producto:

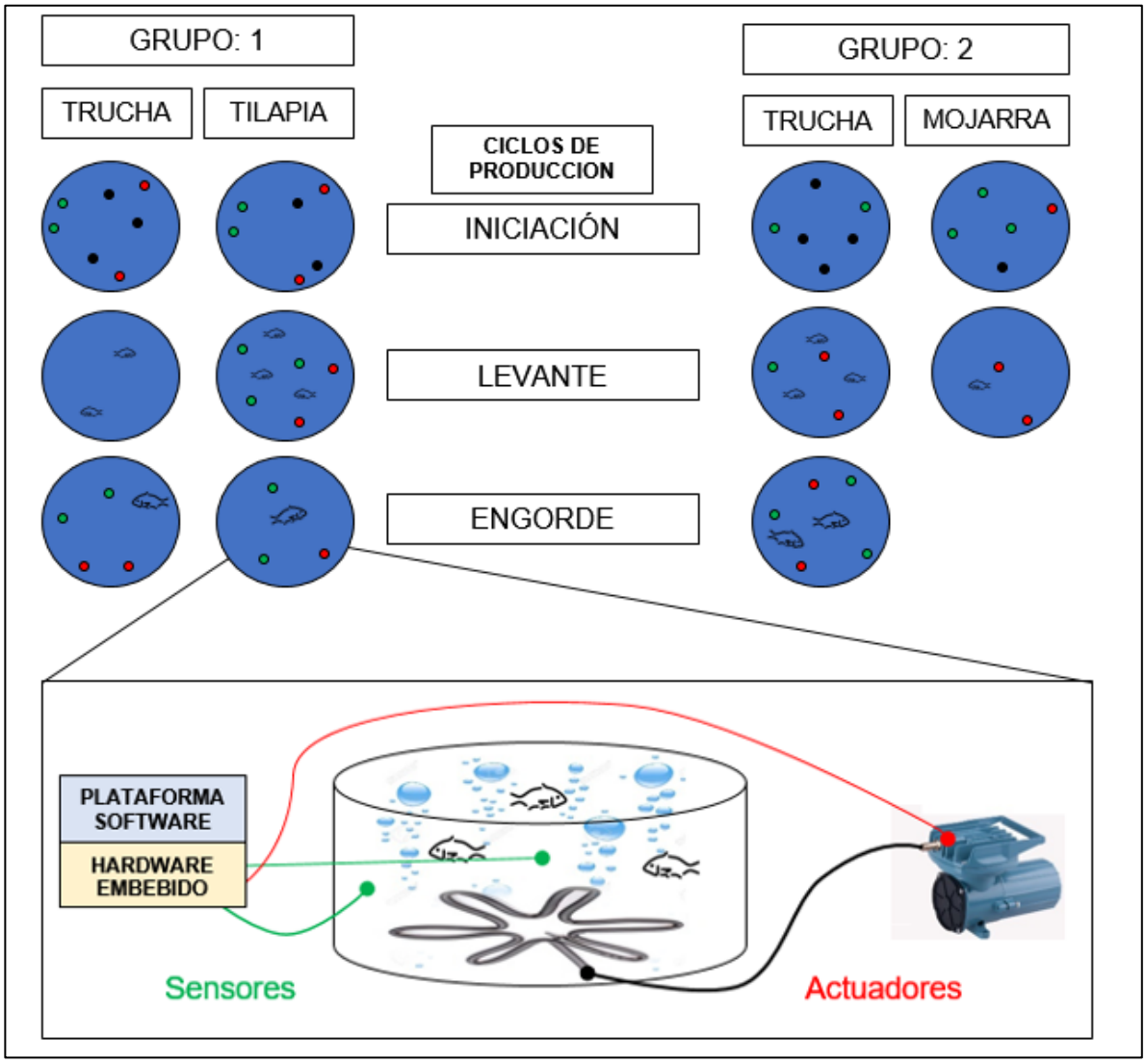
El prototipo final implementado en el proyecto

# 6. DESARROLLO DEL PROYECTO

## 6.1 ENFOQUE DEL PROYECTO

La visión de este proyecto es la de proveer el diseño de una plataforma software que permita apoyar los procesos de crianza específicamente en el área de la piscicultura.

*Figura 2. Enfoque del proyecto*



La plataforma software realizará el monitoreo de un grupo de estanques como se puede observar en la figura 2. Para cada estanque la plataforma inicia la recolección de datos de las características del ambiente, basándose en un grupo de sensores, que son los encargados de capturar las distintas medidas del ambiente piscícola, para luego interpretarlas mediante una colección de reglas específicas que evaluarán dichas mediciones y determinarán las acciones que se verán reflejadas a través de los actuadores cuando el proceso sea automatizado y de otro tipo de acciones cuando el proceso sea semi-automatizado.

La plataforma podrá apoyar varios grupos paralelamente, en donde cada grupo puede tener un número específico de estanques, con diferentes especies en ellos y en diferentes ciclos del proceso de producción y cada estanque puede estar equipado de muchas formas, es decir, tener diferentes tipos y cantidades de sensores y puede tener o no tener actuadores, esto depende en gran medida de factores como, el tipo de especie que se esté cultivando, las condiciones climatológicas que se presenten en un determinado rango de tiempo, en otras palabras, de las necesidades que el usuario final considere pertinentes en cada uno de los escenarios que se le presenten.

## 6.2 IDENTIFICACIÓN DE LAS NECESIDADES

Para lograr el diseño que se planteó anteriormente es primordial identificar las distintas necesidades de la piscicultura y también de la plataforma.

**6.2.1 Necesidades de la piscicultura.** Apoyándose en la documentación que está disponible en lo que concierne a piscicultura y luego de varias reuniones con el co-director del proyecto Carlos Aníbal Vásquez Cardozo conocedor en el campo de la piscicultura y Coordinador Agroindustrial en el Instituto de Proyección Regional y Educación a Distancia (IPRED) se determinaron las siguientes necesidades:

- Recopilación de datos

  El área de la piscicultura trata con organismos vivos y el contexto en el que se desarrolla todo el proceso está expuesto a contaminarse fácilmente debido a que los peces realizan su necesidad fisiológica de la excreción en el mismo ambiente en el que se alimentan y más aún en el que respiran, esto se traduce en una de las problemáticas que más afecta a los piscicultores, conocer las características del agua (niveles de oxígeno, temperatura, turbidez). Por eso es de vital importancia tener disponible los datos exactos y en tiempo real de lo que sucede en el ambiente en el que viven los peces para de esa manera poder tomar acciones inmediatas que afecten positivamente a los peces disminuyendo a futuro la tasa de morbilidad y mortalidad.

- Tratamiento de la información

  Los peces criados en estanques a diferencia de aquellos que se crían en río no cuentan con el beneficio de que el agua se auto oxigene, esto obliga a los piscicultores a utilizar métodos de oxigenación del agua como lo son las pastillas de $O^2$ o recambio del agua que es el más económico actualmente, pero esto repercute en una disminución del recurso gastado inadecuadamente debido al desconocimiento del momento en que se hace realmente necesario suministrar oxígeno al estanque. Además, Colombia tiene fenómenos climáticos durante todo el año (fenómeno del niño, de la niña, etc.) y estos cambios afectan positiva o negativamente el desarrollo de las especies que se estén cultivando; también depende del tipo de pez, ya que los peces en función del clima pueden tener beneficios como aumento de la velocidad del crecimiento y engorde. Por las razones anteriores, se hace necesario realizar un tratamiento a corto plazo de los factores que afectan al sistema, y a futuro un seguimiento (histórico) a los datos recopilados para poder tomar decisiones en un nuevo ciclo de producción, donde los peces se beneficien de las condiciones climáticas y el piscicultor se vea beneficiado económicamente.

- Ejecución de tareas

  Si bien es cierto que no se puede prescindir del factor humano en la realización de las tareas en la piscicultura, hay algunas tareas que se podrían “automatizar y controlar”, para poner un ejemplo, la comida que se suministra a los peces (dependiendo de la etapa de desarrollo en la que se encuentren) siempre se realiza en ciclos periódicos repetitivos y en cantidades controladas, esto con el fin de evitar sobrecostos y sedimentación de la comida que los peces no terminan consumiendo por varias razones. Se hace necesario que algunas tareas diarias en la piscicultura se puedan sistematizar para disminuir el porcentaje de error humano y controlar para evitar sobrecostos innecesarios.

**6.2.2 Necesidades de la arquitectura.** Se describen las siguientes necesidades desde el punto de la arquitectura y las cuales la plataforma propuesta debe implementar:

- Escalabilidad

  En el momento en que se implemente la plataforma, el piscicultor tiene un número fijo de estanques, con los cuales se arranca el sistema y empieza el proceso de monitoreo, pero con el pasar del tiempo y si se dan las condiciones (espacio disponible, recursos económicos) puede aumentar el número de estanques ya sea por nuevas especies a cultivar o por elevar el número de producción de las especies que ya se cultiven. Por las anteriores razones se hace necesario que la plataforma sea escalable y permita agregar los nuevos estanques que también se deseen monitorear.

- Modularidad

  Es necesario que la plataforma sea lo más modular posible, es decir, que todos los módulos que conforman la plataforma sean tan independientes los unos de los otros; esto con el fin de que, si en un determinado momento se

requiere reemplazar un componente ya implementado, este sea sustituido (por uno que tenga mejores características de desempeño y/o ejecución) y no genere mayor traumatismo dentro de la plataforma.

- Abierta

  La plataforma diseñada requiere que sea abierta debido a que debe permitir ajustarse a nuevas necesidades que se puedan presentar en el futuro y que en el momento de desarrollar este proyecto aún no se tienen en cuenta. Al permitir ser abierta, se garantiza que no se vuelva obsoleta al tratar de enfrentarse a nuevos problemas emergentes y así otros desarrolladores puedan beneficiarse y hacer uso de la plataforma.

- Adaptabilidad (plataformas hardware, necesidades específicas de la explotación)

  Se hace necesario que la plataforma sea adaptable, debido a la gran cantidad de plataformas embebidas (Arduino, Intel, LattePanda, Raspberry Pi) que ofrece el mercado actual, algunas placas son más potentes que otras debido a la cantidad de recursos que utilizan y pueden ejecutar en un determinado lapso de tiempo, esto debido a las diferentes necesidades a las que desean darle alguna solución. Ahora que se tienen identificadas las necesidades de la piscicultura, se hace necesario que la plataforma se adapte a dichas necesidades que se especificaron anteriormente teniendo en cuenta las distintas plataformas que existen.

- Extensibilidad

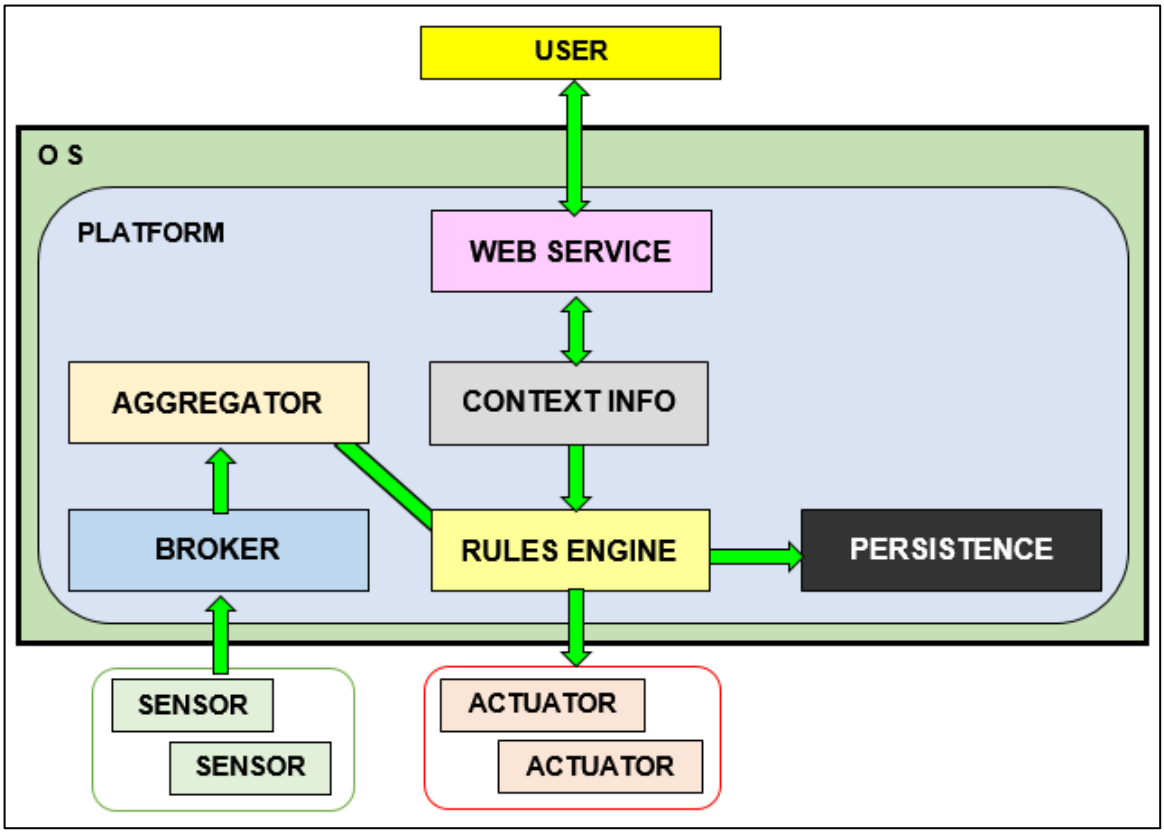
  Debido a la gran versatilidad de sensores (tipos de medición, materiales en que se han diseñado) y actuadores (marcas, finalidad para la cual fueron construidos) que existen son heterogéneos, cada uno de ellos utiliza distintos protocolos de comunicación, ejecución, conectividad física, etc. Se hace

necesario que la plataforma pueda extenderse a las necesidades descritas anteriormente.

## 6.3 ARQUITECTURA SOFTWARE

Con base en las necesidades que se identificaron previamente en el capítulo 6.2, se construyó el diseño de la arquitectura software:

*Figura 3. Diseño de la arquitectura software*




Teniendo definido todo el diseño de la arquitectura es necesario describir cada uno de los módulos que conforman la plataforma.

**6.3.1 Broker.** El broker (o intermediario) es el encargado de recibir y entregar de forma ordenada los datos que son capturados desde los diferentes sensores que se encuentren vinculados al sistema, por medio del método de comunicación

“publish/subscribe”, el cual consiste en que los sensores “publican” (envían) los datos por medio de la función post y el broker los recoge manteniéndolos disponibles y entregandolos a cada uno de los módulos que se suscriban al servicio, garantizando que los datos solo serán entregados a los métodos que los soliciten.

**6.3.2 Aggregator.** Este módulo se encarga de darle un formato específico a los datos que recibe, para ello se apoya en la colección de objetos HashMap que consiste en implementar la interfaz Map y organizar los datos en un patrón llamado “key-value” donde cada uno de los datos recibidos es un “value” y se identifica por una única “key” dentro de la colección. También los tiene disponibles para cuando alguno de los módulos llegue a solicitar su información.

**6.3.3 Web Service.** Este módulo se encarga de comunicarse e interactuar con el usuario final, dándole las alternativas de poder consultar, agregar, modificar o eliminar la información contextual del ambiente, es decir, información que actualmente le es imposible capturar desde los sensores involucrados, algunos ejemplos podrían ser: el tipo de peces que hay en cada estanque, la cantidad de estanques a monitorear actualmente, la fase productiva en la que se encuentra cada estanque, el inicio o la finalización de un ciclo de producción, etc.

**6.3.4 Context information.** Este módulo a diferencia del módulo aggregator se encarga de contener, organizar y mantener disponible la información contextual del ambiente piscícola, es decir, la información que no cambia tan frecuentemente como si lo hacen los datos que capturan los sensores, ejemplos de este tipo de información son: la cantidad de grupos que se encuentren asociados a la plataforma, la cantidad de estanques que se van a monitorear, las especies (o nombres) que se encuentran en cada uno de los estanques, el tipo de clima al que pertenecen los peces, el ciclo de producción (iniciación, levante, engorde) en la que se encuentre actualmente un estanque, etc.

**6.3.5 Rules Engine.** Este módulo apoyado en el framework EasyRules se encarga de contener las reglas que se pueden ejecutar dentro de la plataforma, para ello son necesarios los datos que son entregados por el módulo “aggregator”; teniendo los datos disponibles, las reglas verifican cuales se deben aplicar, debido a que no todas las reglas se aplican con todos los datos, las reglas se enfocan en analizar un tipo de dato específico, un ejemplo sería que las reglas que analizan las mediciones de oxígeno, jamás analizaran las mediciones de tipo temperatura o cualquier otro tipo, esto le da un dinamismo a la plataforma; las reglas analizan que los datos se encuentren en un “rango normal de interacción con el ambiente”, si se presentan anormalidades fuera de los rangos, las reglas ejecutarán unas acciones que apoyadas en la información suministrada por el módulo de “context information” verifican en cual grupo y estanque es necesario realizar una acción de corrección, estas acciones son enviadas a los actuadores para tratar de corregir la alteración.

**6.3.6 Persistence.** Este módulo se encarga de guardar en memoria (persistir) los datos que se han analizado y las acciones que se han creado desde el módulo de “Rules Engine” esto con el fin de tener un respaldo de las órdenes desencadenadas por la plataforma y de los eventos que han tenido lugar durante el tiempo de ejecución del sistema. A largo plazo el objetivo de esta información que se encuentra almacenada, es la de ayudar a tomar decisiones que afecten positivamente el desarrollo de los ciclos productivos de las especies cultivadas.

## 6.4 EXTENSIBILIDAD DE LA PLATAFORMA

**6.4.1 Sensores.** Se definió en la arquitectura un artefacto software conocido como “sensor.tipodelamedicion” (ej. sensor.oxigeno), este se encarga de capturar los datos de los distintos estanques asociados a la plataforma; para implementar cada nuevo tipo de sensor que se desee agregar, es necesario tener en cuenta

una serie de pasos para el perfecto acoplamiento con la plataforma, los cuales son:

- Cada artefacto debe tener mínimo dos clases, la clase llamada “sensor.tipodelamedicion” que se encarga de controlar a cada sensor independientemente y la clase “factory” que se encarga de crear nuevas instancias del sensor utilizando el método “factory-pattern”[15], importante: cada “factory” solo puede crear instancias de un solo tipo de sensor.
- Cada “sensor.tipodelamedicion” debe implementar la interfaz propia: “Sensor” definida dentro de la api de la plataforma la cual tiene 4 métodos:
  -“start” inicia un hilo que repite la ejecución del driver de acuerdo a la necesidad de recolección de los datos asociados al sensor.
  -“stop” detiene la ejecución del driver.
  -“setBus” inyecta la instancia del EventBus a utilizar para la comunicación con la plataforma.
  -“configure” es invocado por el factory para configurar el driver según los parámetros del sensor.
- Cada “sensor.tipodelamedicion” debe implementar la clase “Runnable” que se encarga de ejecutar el método “run” que tiene como objetivo actualizar y capturar los datos de cada sensor.
- Cada “factory” debe implementar la interfaz propia “SensorFactory” definida dentro de la api de la plataforma, la cual tiene 2 métodos:
  -“getSensor” se encarga de recibir los datos que almacena el usuario en un archivo de texto donde define las características necesarias para la creación de nuevos sensores (identificador, tipo, EventBus).
  -“isCapable” verifica que los datos en “getSensor” correspondan al mismo tipo de sensor que factory es capaz de producir, si esto se cumple, se crea una nueva instancia para controlar al nuevo sensor identificado.
- Es necesario establecer en la ubicación: META-INF/services un archivo con la ruta de la interfaz (ej. “uis.brt.sensor.api.SensorFactory”) y dentro del archivo

tendrá una única línea (ej. uis.brt.sensor.oxigeno.Factory) con el nombre de la clase que implementa el patrón factory específico del artefacto.

**6.4.2 Actuadores.** Se definió en la arquitectura un artefacto software denominado como “actuador.nombredelactuador” (ej. actuador.valvula), este se encarga de ejecutar las acciones que son creadas en el módulo de “rules engine” permitiendo que la plataforma pueda automatizar tareas, siempre y cuando tenga un actuador físico conectado a la plataforma. para implementar cada nuevo actuador, es necesario tener en cuenta las siguientes especificaciones:

- Cada artefacto debe tener mínimo dos clases, la clase llamada “actuador.nombredelactuador” que se encarga de controlar a cada actuador independientemente y la clase “factory” que se encarga de crear nuevas instancias del actuador, importante: cada “factory” solo puede crear instancias de un solo tipo de actuador.
- Cada “actuador.nombredelactuador” debe implementar la interfaz propia: “Actuator” definida dentro de la api de la plataforma la cual tiene 5 métodos:
  -“configure” es llamado por el factory y configura el actuador
  -“state” establece el estado de acción del actuador
  -“open” activa la finalidad del actuador.
  -“close” detiene la finalidad del actuador.
  -“execute” se ejecuta dependiendo del estado del actuador y las acciones enviadas desde el motor de reglas.
- Cada clase “factory” debe implementar la interfaz propia “ActuatorFactory” definida dentro de la api de la plataforma, la cual tiene 2 métodos:
  -“getActuator” se encarga de recibir los datos que almacena el usuario en un archivo de texto donde define las características necesarias para la creación de nuevos actuadores (identificador, tipo, estado).

-”isCapable” verifica que los datos en “getActuator” correspondan al mismo tipo de actuador que factory es capaz de producir, si esto se cumple, se crea una nueva instancia para controlar al nuevo actuador identificado.

- Es necesario establecer en la ubicación: META-INF/services un archivo con la ruta de la interfaz (ej. “uis.brt.sensor.api.ActuatorFactory”) y dentro del archivo tendrá una única línea (ej. uis.brt.actuator.valvula.Factory) con el nombre de la clase que implementa el patrón factory específico del artefacto.

**6.4.3 Reglas.** Las reglas a diferencia de los sensores y actuadores son un poco más permisivas debido al hecho de que pueden ser tan sencillas como comparar un valor y emitir un mensaje, o tan complejas como evaluar varios parámetros al tiempo y ejecutar distintas acciones de acuerdo a los resultados obtenidos, o como lo desee el usuario, todo depende de cual sea el fin que se les desee dar; para su implementación es necesario las siguientes instrucciones:

- Cada regla a agregar debe implementar la interfaz “PlatformRule” que gestiona un único método “setData” que se encarga de actualizar la información que la regla necesita para ejecutarse.
- Las reglas se basan en el framework “EasyRules” el cual indica que para la ejecución de acciones es necesario la implementación de 2 métodos:

  -“Evaluate” este método se basa en unas condiciones previamente definidas por el usuario en las cuales el dato que se capta del ambiente debe ser evaluado emitiendo un concepto de verdadero o falso.

  -“Execute” la única manera de que este método se ejecute es si el método anterior emite un concepto verdadero. Este método crea las acciones necesarias que se deben ejecutar en los estanques que presenten algún tipo de alteración fuera de las reglas definidas en “Evaluate”.

# 7. IMPLEMENTACIÓN DEL PROTOTIPO

## 7.1 PLATAFORMA DE EJECUCIÓN

**7.1.1 Placa Intel® Galileo.** La plataforma está diseñada para trabajar sobre arquitecturas embebidas (Raspberry Pi, Arduino) debido a su enfoque con IoT, pero para la construcción del prototipo se debe escoger alguna, así que se decidió trabajar sobre la placa Intel® Galileo ya que en su momento el director del proyecto la tenía disponible y la conocía de antemano por proyectos desarrollados con anterioridad, ya que dicha placa cubriría las necesidades requeridas para la puesta en marcha y pruebas a las que daría lugar el proyecto.

*Figura 4. Placa Intel® Galileo*

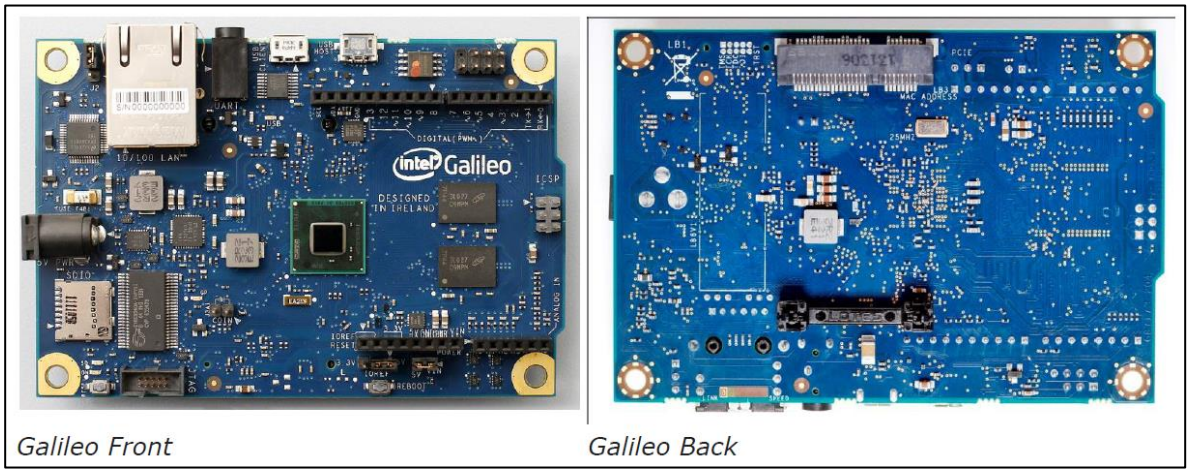



*Tabla 2. Especificaciones de la Placa Intel® Galileo*

| | |
|---|---|
| Procesador | Intel® Quark SoC X1000 |
| Arquitectura | 32 bits Intel Pentium – 400 MHz |
| DRAM | 256 MByte |
| Almacenamiento | Interno: 8 Mbyte<br>Externo: tipo micro SD (limite 32 GByte) |
| Puertos | 1 MiniUSB 2.0 para host |

| | |
|---|---|
| | 1 MiniUSB 2.0 para client<br>1 Ethernet RJ45 10/100 |
| Conectores | 6 pins análogos y 14 pins digitales (Input/Output) |

Fuente: Oficial de Intel®. Galileo Datasheet [16]

**7.1.2 Sensores.** Se listan los sensores que pueden ser una solución en el área de la piscicultura

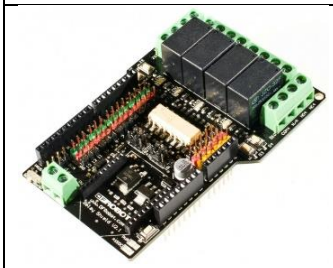
*Tabla 3. Sensores relacionados con la piscicultura*

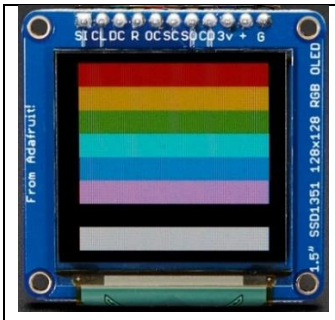
| IMAGEN | MEDICIÓN | DESCRIPCIÓN |
|---|---|---|
|  | *Figura 5. TEMPERATURA* | Mide la temperatura del agua en un rango de -55°C y 125°C. [17] |
|  | *Figura 6. OXIGENO* | Mide la cantidad de oxígeno disuelto en el agua en un rango de 0.1 y 100% de saturación. [18] |
|  | *Figura 7. PH* | Mide el pH del agua entre 0 y 14 con una precisión de ±0.1pH. [19] |
|  | *Figura 8. TURBIDEZ* | Mide la calidad del agua mediante la luz analizando la cantidad total de solidos suspendidos en el líquido. [20] |

| | | |
|---|---|---|
|  | *Figura 9. ORP* | Mide la calidad del agua analizando el Potencial de Reducción de la Oxidación en el líquido. [21] |
|  | *Figura 10. NIVEL* | Mide el nivel del agua sin la necesidad de entrar en contacto con el líquido alargando su vida útil. [22] |

**7.1.3 Actuadores.** Se listan los actuadores que pueden ser una solución en el área de la piscicultura

*Tabla 4. Actuadores relacionados con la piscicultura*

| IMAGEN | EJECUCIÓN | DESCRIPCIÓN |
|---|---|---|
|  | *Figura 11. INTERRUPTOR* | Permite controlar (Apertura/Cierre) puertas de contención, bombas de suministro de agua, etc. Utilizado en robótica y domótica. [23] |
|  | *Figura 12. VÁLVULA* | Permite controlar el flujo del agua. Trabaja con voltaje de 12V. [24] |

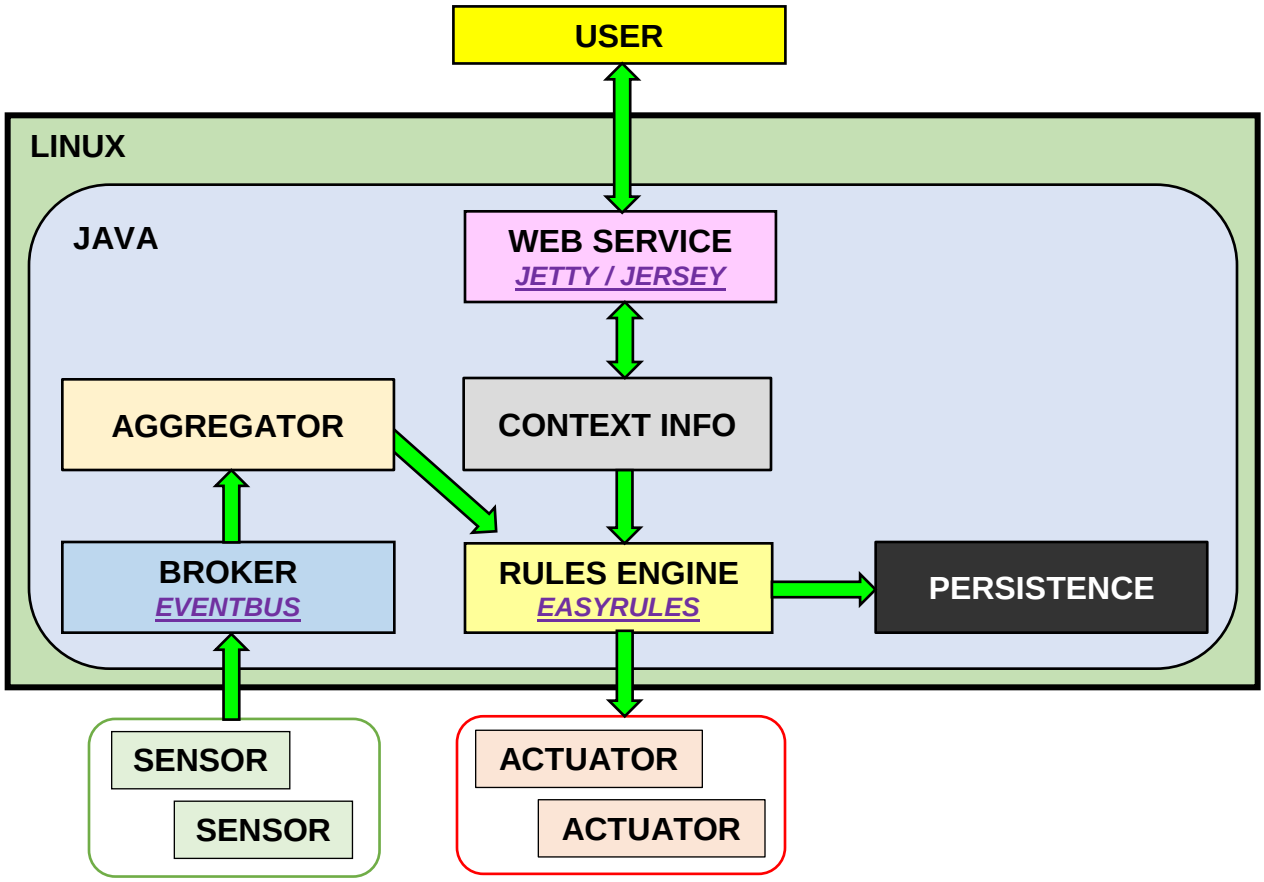
| | | |
|---|---|---|
|  | *Figura 13. PANTALLA* | Permite visualizar en una pantalla OLED de 1.5 pulgadas y 16 bits la información que se ejecute dentro del sistema. [25] |
|  | *Figura 14. BLUETOOTH* | Permite el intercambio de información en un rango de 70 metros. [26] |

Existen empresas [27] dedicadas a la creación, adaptación y distribución de una gran gama de sensores y actuadores para un gran número de placas (Intel, Arduino, Raspberry Pi).

**7.1.4 Sistema Operativo: Yocto Project.** El proyecto Yocto es un proyecto de colaboración de código abierto que proporciona plantillas, herramientas y métodos que ayudan a crear sistemas personalizados basados en Linux para productos embebidos, independientemente de la arquitectura del hardware. Fue fundada en 2010 como una colaboración entre muchos fabricantes de hardware, proveedores de sistemas operativos de código abierto y empresas de electrónica para poner algo de orden en el caos del desarrollo de “Linux embebido”. Intel es miembro fundador del Proyecto Yocto y ha sido miembro Gold desde 2011. [28]

## 7.2 IMPLEMENTACIÓN DEL SOFTWARE

*Figura 15. Arquitectura software y frameworks implementados*




Se puede observar en la figura anterior, las distintas herramientas y frameworks que fueron utilizados a la hora de ejecutar la implementación software, cabe aclarar que módulos como el Broker y Sensor se basaron y adaptaron de un proyecto existente[8], todos los demás módulos fueron desarrollados desde cero, a continuación, se definirá cada herramienta utilizada y su función dentro de la plataforma.

**7.2.1 Java.** Se escogió como lenguaje de programación por ser orientado objetos, multiplataforma y más específicamente por ser portable y debido a que existe una tendencia a que los sistemas embebidos sean desarrollados con java, se pueden encontrar frameworks basados en este lenguaje y de esta manera poder integrarlos a la plataforma.

**7.2.2 EventBus.** Se escogió para que apoye el módulo “broker” fue desarrollado por Google y permite la comunicación utilizando el estilo “publish-subscribe” entre los componentes sin requerir que estos se registren explícitamente entre sí (y así estar conscientes el uno del otro). Está diseñado exclusivamente para reemplazar la distribución de eventos Java en el proceso tradicional mediante el registro explícito. Distribuye los eventos a los oyentes y proporciona formas para que los oyentes se registren. [29]

**7.2.3 Easyrules.** Es un motor de reglas y se escogió para que apoye al módulo de “Rules Engine” su desarrollo está basado en POJO (Plain Old Java Object), es decir que utiliza clases simples y no necesita de ningún framework especial. muy simple y liviano pero potente debido a que utiliza abstracciones para definir las reglas de juego y aplicarlas fácilmente. En otras palabras, permite construir y administrar una “colección de reglas” que se componen de objetos que contienen unas condiciones y acciones, que están disponibles para cuando el usuario las solicite, se evalúen con los parámetros que reciban dichas reglas y ejecutar unas acciones de ser necesario. [30]

**7.2.4 Jetty.** Es un proyecto de software libre desarrollado en java, sencillo y eficiente que permite montar servidores HTTP y un contenedor de servlets que lo hace ideal para integrarse con plataformas embebidas que no necesitan demasiados recursos (livianas) de ejecución, razón por los cual se ha escogido para ser parte de este proyecto. [31]

**7.2.5 Jersey.** Es un framework de servicios web RESTful de código abierto desarrollado en java que principalmente da soporte a la API JAX-RS, ampliando las herramientas de esta con funciones y utilidades adicionales para simplificar aún más el servicio RESTful potenciando la comunicación cliente-servidor y el desarrollo del lado del cliente.

### 7.2.6 Herramientas de soporte

- GitHub

  Permite llevar un control centralizado y organizado de las diferentes versiones que el proyecto fuera generando. Ideal ya que es una herramienta que comprueba los cambios generados en la programación, los actualiza por medio del servicio en la nube y notifica a todos aquellos que estén involucrados en el proyecto. Se escogió esta herramienta para tener una comunicación más flexible con el director del proyecto, porque nos permitió acceder al código en cualquier momento y lugar, llevando un control de los cambios que generará cualquiera de las partes.

- Maven

  Esta herramienta permite gestionar librerías, actualizar, empaquetar y probar los proyectos con los cuales se esté trabajando, procesos que en la práctica restan demasiado tiempo al desarrollador. Es de código abierto y maneja un estricto ciclo de vida lineal que garantiza que cualquier etapa que se ejecute realizará las etapas preliminares para su correcta ejecución.

- HTML

  Es el lenguaje por excelencia utilizado por los navegadores para mostrar las páginas web, permitiendo la visualización, intercambio y manipulación de la información dentro del sistema, siendo un gran aliado para los desarrolladores

en cuanto a creaciones de interfaces de comunicación con los usuarios finales.

## 7.3 ELABORACIÓN DEL PROTOTIPO

Se procede a construir un prototipo funcional que permita realizar la última fase de pruebas en un ambiente real, pero aún más importante, validar el diseño que se planteó como posible solución a la problemática actual y que apoye los procesos de crianza en la piscicultura.

**7.3.1 Componentes hardware del prototipo.** A continuación, se listan los recursos con los cuales se construirá el prototipo:

- 1 placa Intel® Galileo (con adaptador de corriente)
- 1 protoboard
- 1 memoria microSD de 8 GByte
- 1.5 metros de cable UTP Categoría 5e
- 1 sensor de temperatura
- 1 computador portátil

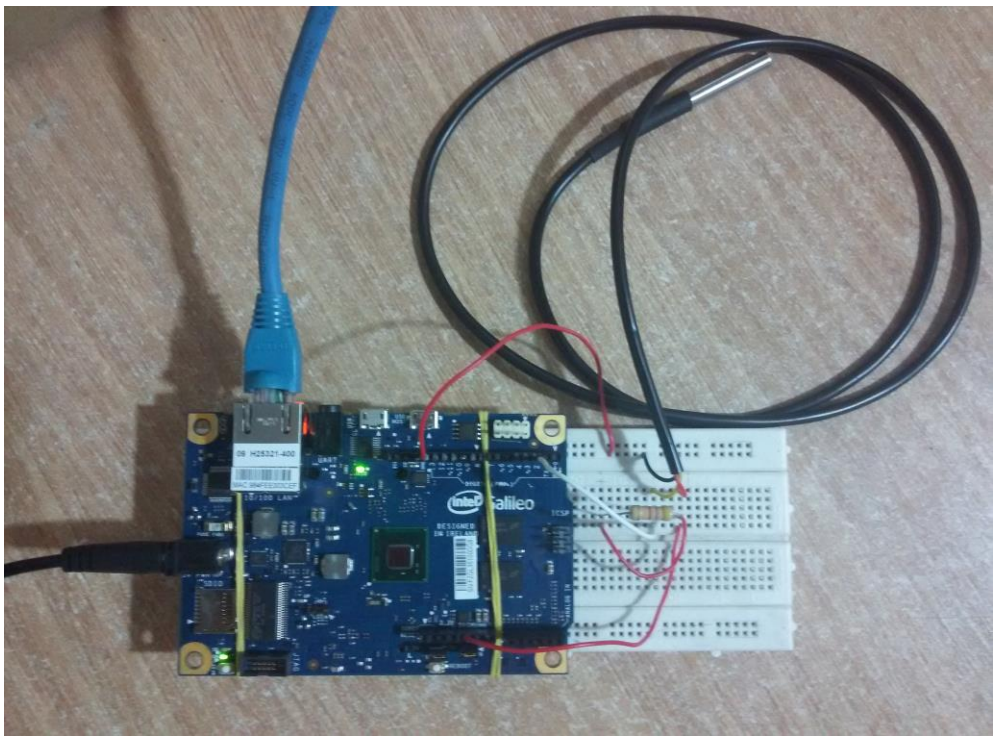
**7.3.2 Ambientación tecnológica.** Conociendo las características físicas de los recursos que se van a implementar, se hizo necesario realizar una serie de configuraciones previas para que el hardware estuviera en óptimas condiciones y estuviese lo más estable posible al momento de ejecutar este proyecto.

Teniendo físicamente la placa Intel® Galileo fue necesario actualizar sus controladores, firmware, etc, ya que era una placa nueva y para lo cual se utilizó la “Getting Started Guide” [32] con la cual fue posible probar su correcto funcionamiento, realizar algunos ejemplos para familiarizarse y ya entrando en materia, se hizo necesario instalar un Sistema Operativo para trabajar con

sistemas embebidos, para lo cual se utilizó el sitio “Getting Started with the Intel® Galileo Board on Windows” [33] con el cual fue posible descargar un SO booteable (en este caso yocto project) en una microSD, conectarse por el puerto Ethernet del portátil a la placa intel utilizando el programa putty (o GitHub) accediendo por medio de la dirección IP que el computador host asigno automaticamente por el servicio DHCP, para conocer dicha dirección existen programas gratuitos para capturar el tráfico entre los dispositivos y conocer las respectivas IP´s, algunos de ellos son: “Angry IP Scanner”, “Wireshark” o dado el caso, si se conectó la tarjeta al modem WIFI hay una app móvil llamada “Fing” que permite ver las direcciones asignadas a los dispositivos conectados a la red junto a la dirección MAC.

Se realizan las respectivas conexiones y obtenemos una interconexión de todos los componentes involucrados para la ejecución del prototipo:

*Figura 16. Construcción física del prototipo*



**7.3.3 Empaquetado del prototipo.** A este punto del proyecto ya se cuenta con una versión lo suficientemente apropiada de la plataforma, que fue programada en JAVA y se encuentra disponible en el repositorio de GitHub [34] utilizando el IDE (Integrated Development Environment) Eclipse y con la ayuda de la herramienta Maven procedemos a empaquetar el core, los sensores y actuadores en sus respectivos ".jar". Esto nos permitirá desplegar el prototipo dentro de la placa Intel galileo.

# 8. VALIDACIÓN

A continuación, se documenta el procedimiento llevado en cada una de las pruebas que se realizaron sobre el prototipo hardware construido en el capítulo 7.3 para validar la plataforma software diseñada durante la ejecución del proyecto.

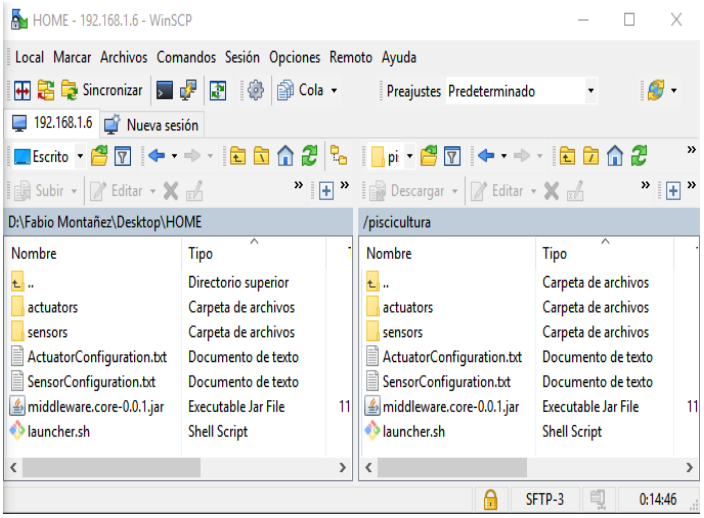
## 8.1 Pruebas de despliegue

Los objetivos de esta prueba son los siguientes:

- Instalar los paquetes .jar en la arquitectura embebida escogida
- Desplegar la plataforma software
- verificar el acoplamiento entre la plataforma y la arquitectura

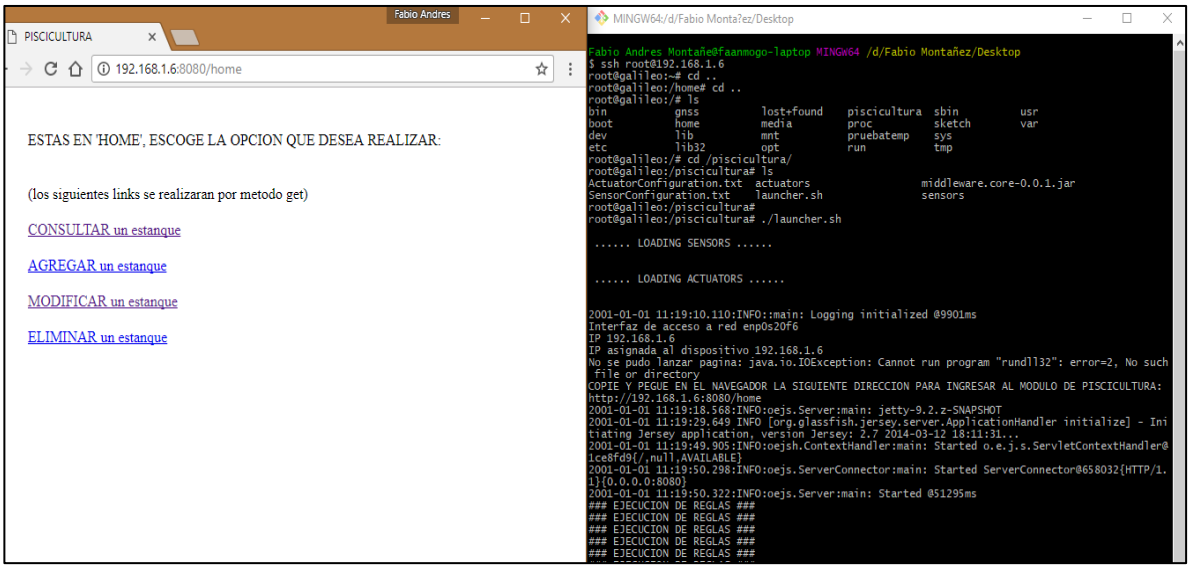
Teniendo la placa conectada a la fuente de alimentación, accedemos a ella por conexión SSH, por medio de la IP asignada (en este momento 192.168.1.6). Para instalar (o copiar) los archivos de la plataforma utilizamos el programa “WinSCP” (debe ser instalado en el equipo Host, en este caso Windows) que nos permite trabajar sobre la típica interfaz de carpetas de Windows, buscamos los archivos en el equipo Host y los pasamos a la placa Embebida en una nueva carpeta llamada piscicultura en la raíz de los archivos, como se muestra en la figura 17

*Figura 17. Copia de los paquetes de la plataforma a Linux*



Ahora que tenemos los archivos en el placa, ingresamos a ella por medio de la consola de comandos Git Bash (de GitHub) y la instrucción: “ssh root@192.168.1.6” Ahora debemos estar dentro del sistema de archivos de la placa en el directorio “root”, navegamos dentro de los diferentes directorios con el comando “cd ..” y mostramos su contenido con “ls” buscando la carpeta recién agregada al sistema piscicultura y dentro de ella nos disponemos a desplegar la plataforma por medio del script “launcher.sh” y la instrucción: “./launcher.sh” ejecutamos con enter y obtenemos los siguientes resultados:

*Figura 18. Arranque y despliegue de la plataforma en el sistema embebido*




Podemos observar en la figura 18 que la plataforma arranca e inicia con la detección automática de los sensores y actuadores que estén asociados a la plataforma, sigue iniciando los otros servicios, como el servidor Jetty y el servicio Jersey y al final vemos una instrucción “### EJECUCION DE REGLAS ###” que nos indica que está lanzando el servicio de análisis de las reglas que estén dentro de la plataforma una y otra vez cada cierto periodo de tiempo. Cabe aclarar que no encuentra sensores ni actuadores ni reglas porque estas pruebas se realizarán en la siguiente sección. También se verifica que podemos acceder a la interfaz web

por medio del navegador y la dirección “http://192.168.1.6:8080/home” y podemos acceder en las distintas opciones que nos brinda la interfaz.

Se observa que la plataforma lanzó una excepción: “No se pudo lanzar página: java.io.IOException: Cannot run program "rundll32": error=2, No such file or directory” esto se debe a que el programa “rundll32” es específico del SO Windows y se encarga de lanzar automáticamente la interfaz web, pero vemos que no es impedimento para acceder al servicio web.

Podemos decir que el sistema embebido y la plataforma se acoplaron adecuadamente y la prueba de despliegue fue exitosa.

### 8.2 Pruebas de extensión y pruebas funcionales

Los objetivos de esta prueba son validar el acoplamiento y el funcionamiento de nuevos componentes como lo son:

- Sensores
- Actuadores
- Reglas

Siguiendo en el punto donde dejamos las pruebas de despliegue, procedemos a agregar un sensor de temperatura físico, a construir y agregar una regla que se asocie con el sensor acoplado recientemente y un actuador simulado que nos permita visualizar las acciones que se generaron desde la regla construida.

Es importante aclarar que la plataforma fue diseñada para analizar los datos en el ambiente de la piscicultura, por eso, adicionalmente fue necesario ingresar la siguiente información:

INFORMACIÓN DEL CONTEXTO:

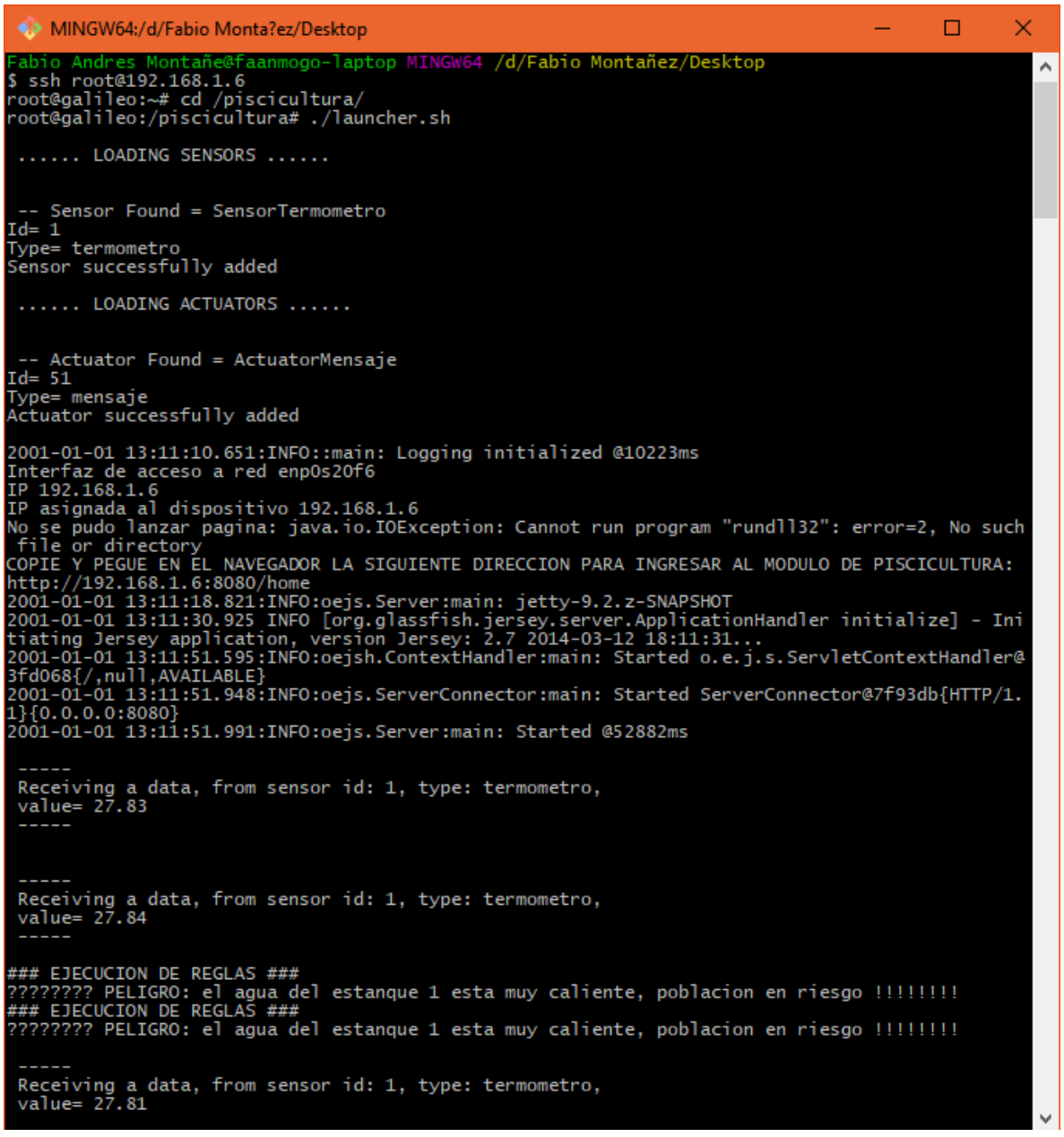
- NÚMERO DE GRUPO: 1
- NÚMERO ESTANQUE: 1
- NOMBRE DEL PEZ: trucha
- TIPO DE CLIMA: frío
- CICLO PRODUCTIVO: iniciación
  SENSORES Y ACTUADORES DEL ESTANQUE
- ID: 1: termómetro
- ID: 51: mensaje

A esta información se le conoce como “información contextual” y se guarda en el módulo de “context information” y su función es situar al sistema en un escenario piscícola. Es decir, de nada sirve analizar los datos de los sensores si no se conoce el estanque al que se le van a aplicar las acciones, o los actuadores que se deben activar o cuales estanques son los que tienen los sensores que capturan los datos. Es por eso que es necesario, para ver los resultados de esta prueba, al menos un estanque con las características definidas anteriormente.

Procedemos a conectar físicamente un sensor de temperatura como se puede apreciar en la figura 16 a codificarlo y agregarlo a la plataforma empaquetado en un “.jar” en la carpeta denominada “sensors”, y de la misma manera con el actuador simulado de nombre “mensaje” y lo agregamos a la carpeta “actuators”. Dentro de los archivos “SensorConfiguration.txt” y “ActuatorConfiguration.txt” la información correspondiente a las caracteristicas del sensor y del actuador respectivamente. Por último, agregamos una regla de nombre “pezfrio”, esta regla se encarga de analizar los datos que el sensor capture del ambiente y determinar: si la medición es menor a 20° C, no hay problema, no se crean acciones, pero, si la medición es mayor, nos cree una acción donde nos muestre por medio del actuador “mensaje” una advertencia de que la temperatura no es apta para las

especies de clima frío. Procedemos a lanzar la plataforma obteniendo los siguientes resultados:

*Figura 19. Pruebas de extensión y pruebas funcionales*

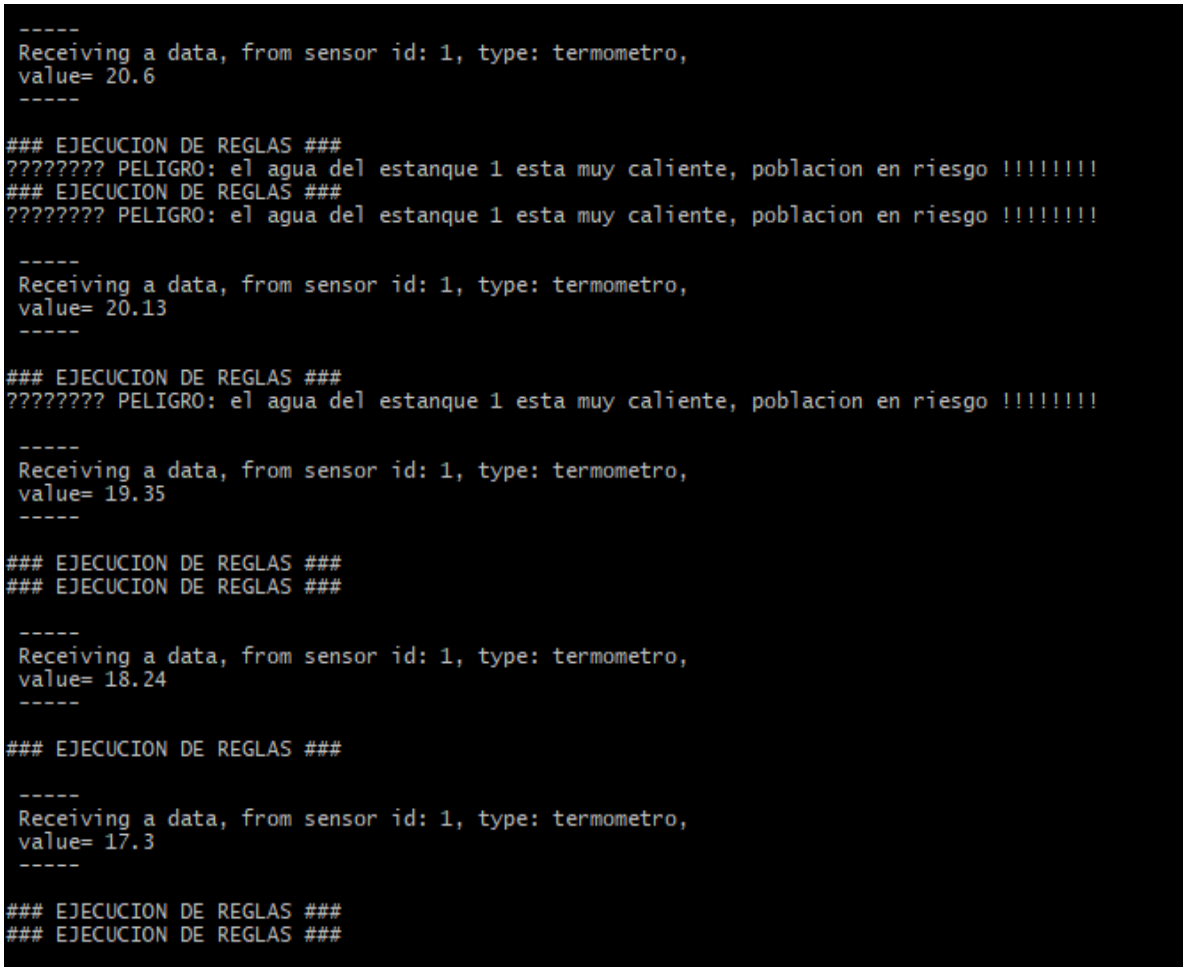



Podemos verificar en la parte superior de la figura 19 que el sensor de temperatura y el actuador de mensajes fueron encontrados y agregados

satisfactoriamente por la plataforma y en la parte de abajo se puede observar que la regla “pezfrio” está enviando un mensaje: “???????? PELIGRO: ¡el agua del estanque 1 está muy caliente, población en riesgo !!!!!!!!”.

Luego forzamos al sensor a capturar temperaturas frías (acercando hielo al sensor) y luego de que mide valores menores a 20° C (ideales para especies de clima frío) el mensaje de advertencia deja de aparecer como se evidencia en la figura 20, esto es debido a que la regla solo se activa para temperaturas altas y la información contextual que se ingresó a la plataforma contemplaba para el estanque #1 peces de clima frío, en este caso trucha en ciclo de iniciación.

*Figura 20. Pruebas de ejecución de reglas*



De esta prueba podemos decir que:

- La plataforma es extensible debido a que permite el acoplamiento de nuevos actuadores y sensores y se comprobó su correcto funcionamiento capturando mediciones y ejecutando acciones.
- Permite agregar nuevas reglas y configurarlas según las necesidades a controlar.
- Las reglas se están aplicando correctamente, de acuerdo a los parámetros que se le configuraron, permitiendo afirmar que las pruebas funcionales fueron exitosas.

### 8.3 Escenario de prueba

Junto al co-director Carlos Aníbal Vásquez Cardozo se planteó un escenario típico que se presenta en la piscicultura con el fin de validar la plataforma desde aspectos más cercanos a la realidad. El escenario planteado es el siguiente:

*Tabla 5. Escenario de prueba contextual*

| | **ESTANQUES** | | | |
|---|---|---|---|---|
| | **#1** | **#2** | **#3** | **#4** |
| **GRUPO** | 1 | 2 | 1 | 2 |
| **PEZ** | trucha | mojarra | cachama | bagre |
| **CLIMA** | frio | calido | calido | calido |
| **CICLO** | engorde | levante | iniciacion | engorde |
| | **SENSORES** | | | |
| **ID MEDICIÓN** | 5 temperatura | 2 ph | | 8 oxigeno |
| **ID MEDICIÓN** | 3 temperatura | 6 oxigeno | | 1 temperatura |
| | **ACTUADORES** | | | |
| **ID TIPO** | 56 mensaje | 51 mensaje | 55 valvula | 53 valvula |

| | | | | |
|---|---|---|---|---|
| **ID** | 59 | 54 | 57 | 52 |
| **TIPO** | valvula | valvula | mensaje | mensaje |

Se procede a codificar y empaquetar el sensor de oxígeno y el actuador de tipo válvula de la misma manera como se realizó en la prueba 8.2 con el sensor de temperatura y el actuador de mensaje.

Adicionalmente se agregarán las siguientes reglas:

- “Pezfrio”

  Esta regla se encarga de crear acciones si evalúa mediciones mayores o iguales a 20° C. Sensor asociado: temperatura. Actuador asociado: mensaje.
- “Pezcalido”

  Esta regla se encarga de crear acciones si evalúa mediciones por debajo de 20° C. Sensor asociado: temperatura. Actuador asociado: mensaje.
- “Oxigenobajo”

  Esta regla se encarga de crear acciones si evalúa oxigeno menor o igual a 4 ppm. Sensor asociado: oxigeno. actuador asociado: válvula.
- “Oxigenoalto”

  Esta regla se encarga de crear acciones si evalúa oxigeno mayor o igual a 5 ppm. Sensor asociado: oxigeno. actuador asociado: válvula.

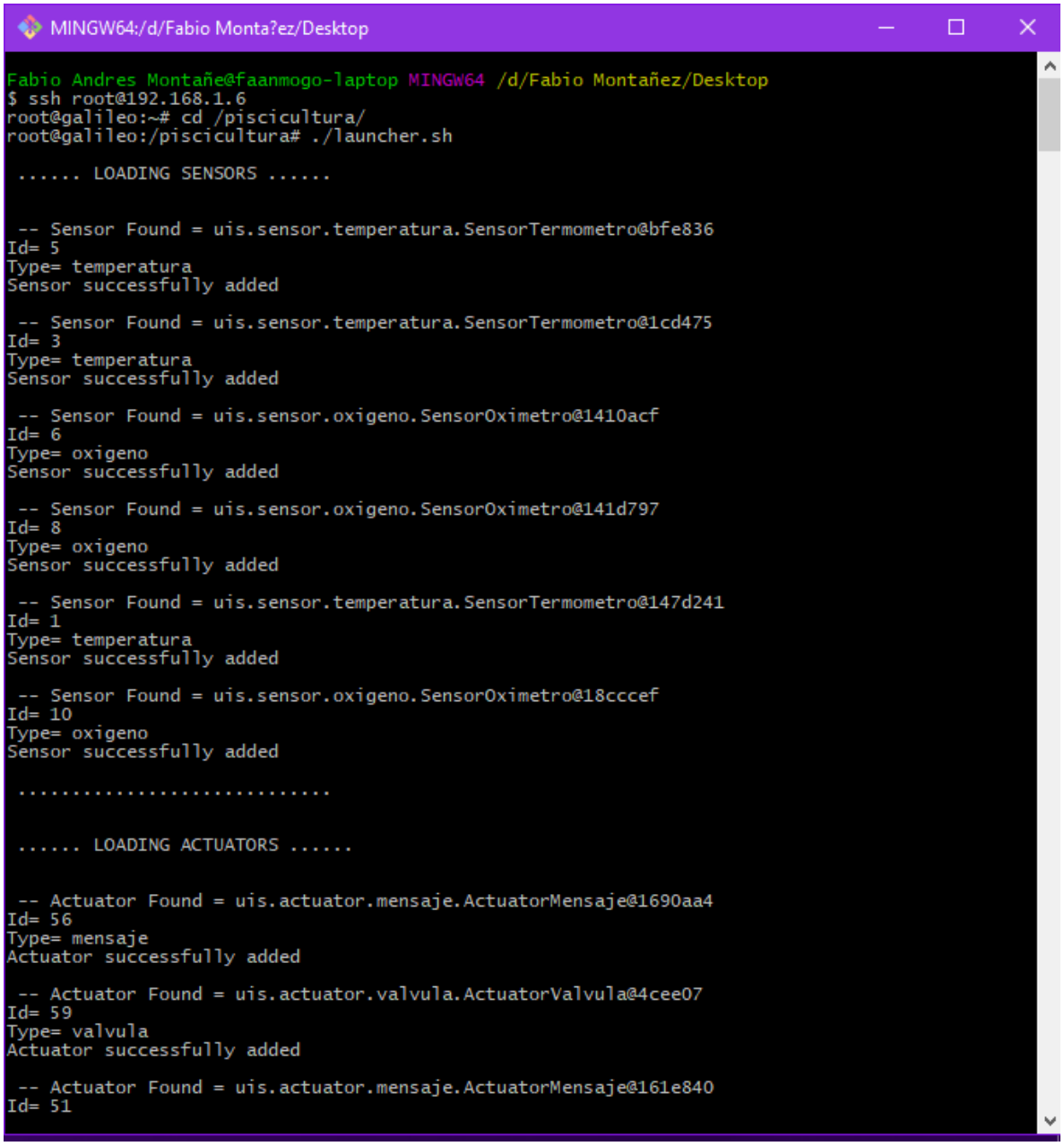
A tener en cuenta:

- De todos los sensores agregados el único que se implementó físicamente y capturará información real de la temperatura será el que se encuentra en el estanque #1 con id: 5. Todos los demás serán valores simulados.
- En el estanque #2 se puso un sensor de ph con id: 2. Esto se hizo intencionalmente para mostrar que sucede con los sensores que se ingresan contextualmente pero no se codifican ni empaquetan en la plataforma.

- En el archivo “SensorConfiguration.txt” se agregó intencionalmente un sensor de oxigeno con id: 10, para mostrar que sucede con los sensores que se ingresan a la plataforma, pero no se asocian contextualmente a ningún estanque.
- Todos los actuadores utilizados en esta prueba son simulados.

Procedemos a montar el prototipo en la placa Intel Galileo, ejecutamos y obtenemos los siguientes resultados:

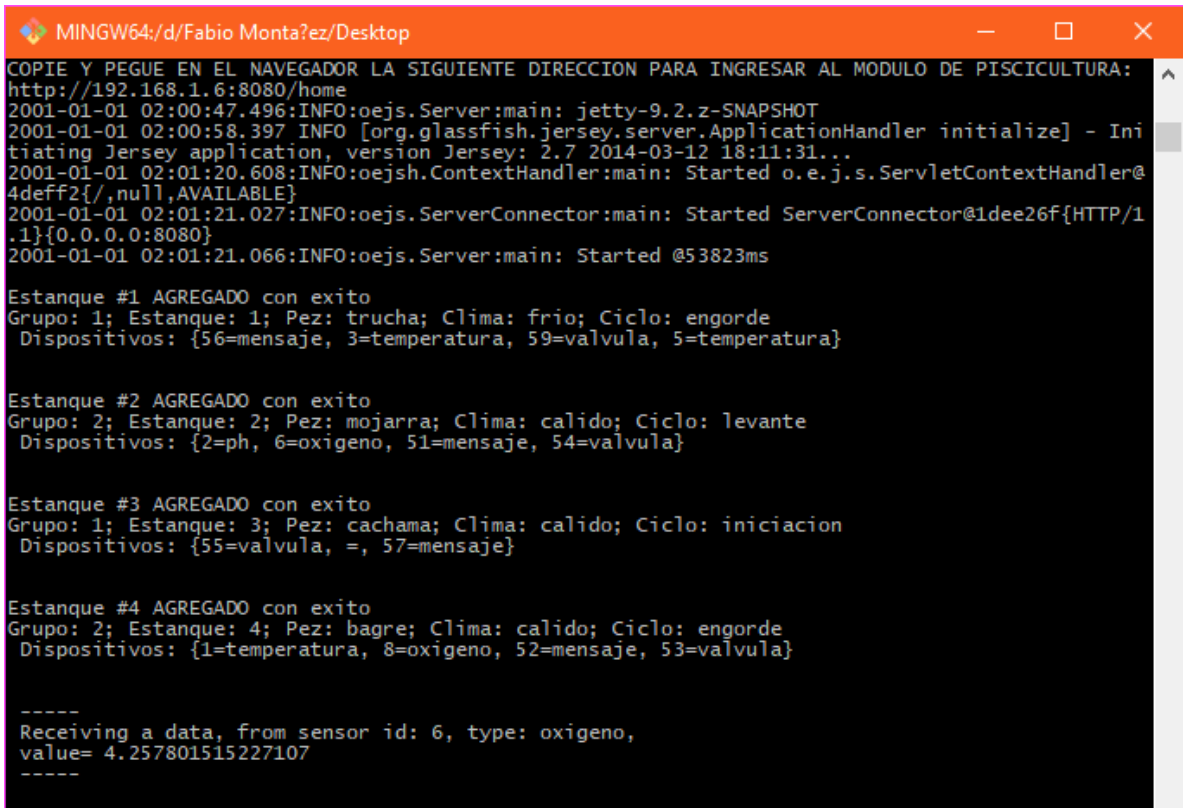
*Figura 21. Reconocimiento de diferentes tipos de sensores y actuadores*




Podemos evidenciar que se identificaron satisfactoriamente los sensores y actuadores ingresados a la plataforma.

Cada vez que se ingresa un nuevo estanque al sistema la plataforma nos notifica. Para esta prueba podemos ver en la figura 22. la agregación de los 4 estanques definidos en la tabla x

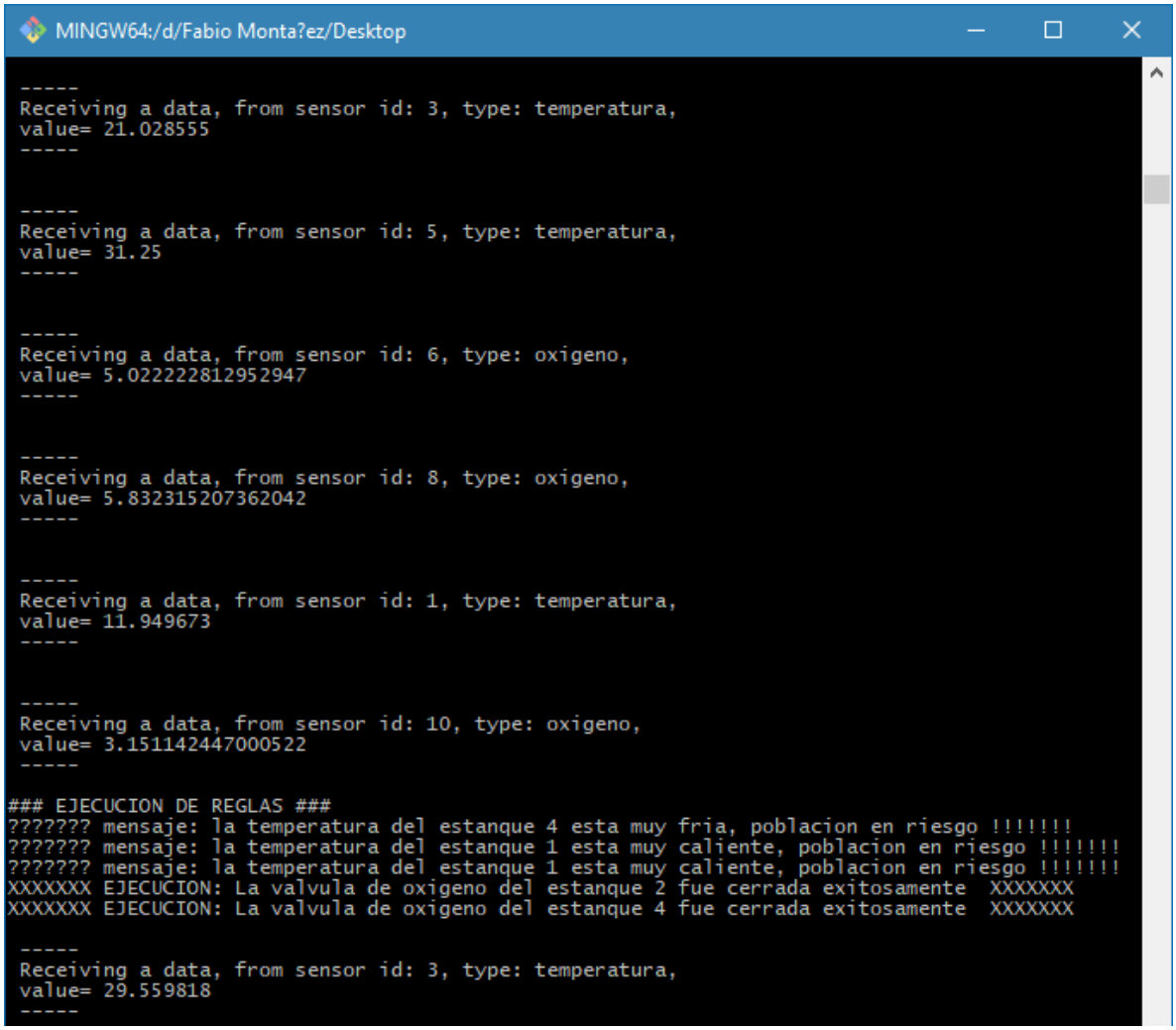
*Figura 22. Agregación de distintos contextos (estanques)*




La plataforma comienza el proceso de captura de las distintas mediciones de los sensores involucrados y luego estos son analizados por las reglas que se registraron al inicio de la prueba. Se puede validar en la figura 23. las siguientes apreciaciones:

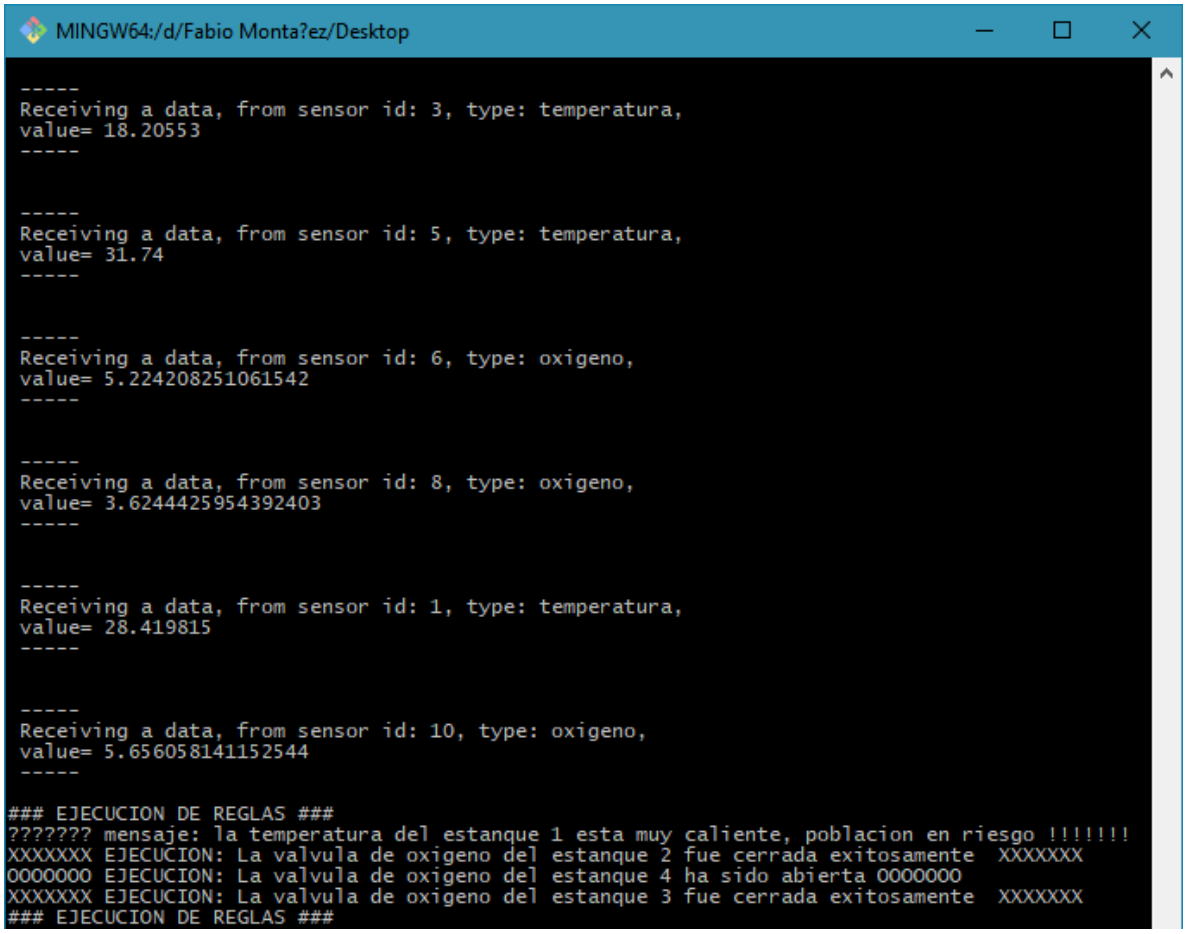
- Se emiten 2 mensajes para el estanque #1 debido a que en un principio este estanque tenia configurados dos sensores de temperatura: el sensor físico (id: 5) y el sensor simulado (id: 3).
- Se emite para el estanque #4, un mensaje (actuador id: 52) de advertencia por la medición de temperatura (sensor id:1) y una acción (actuador id: 53) de tipo válvula por la medición de oxigeno (sensor id: 8).

*Figura 23. captura de distintas mediciones y ejecución de reglas*




Ahora modificamos contextualmente el estanque #3 recordando que este estanque no tenía ningún sensor asociado desde el inicio de la prueba, por lo tanto, no tendría ningún tipo de ejecución en las reglas debido a su falta de datos, por esta misma razón no se emitió ninguna acción para este estanque en la figura 23. Es aquí donde entra a tomar importancia el sensor de oxigeno con id: 10, que no estaba asociado a ningún contexto, pero estaba capturando datos del ambiente, se lo agregamos al estanque #3 guardamos cambios y se puede comprobar en la figura 24. que la plataforma está analizando las reglas y creando acciones para este estanque.

*Figura 24. Agregación de dispositivos en el contexto*



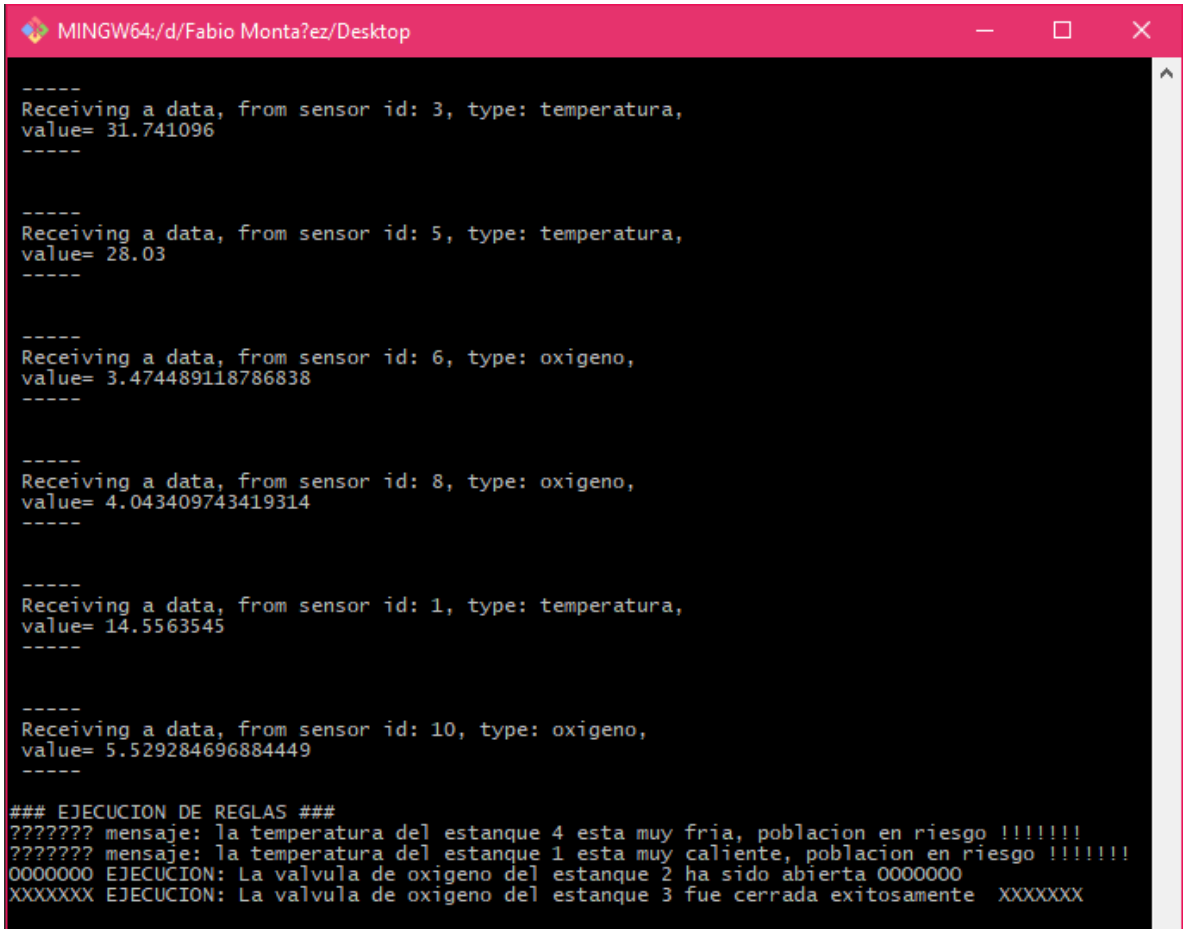

Ahora se modificarán los contextos eliminando sensores y actuadores:

*Tabla 6. Sensores y actuadores eliminados del contexto*

| | **ESTANQUES** | | | |
|---|---|---|---|---|
| | **#1** | **#2** | **#3** | **#4** |
| | SENSORES | | | |
| **ID MEDICIÓN** | 5 temperatura | 2 ph | 10 oxigeno | 8 oxigeno |
| **ID MEDICIÓN** | 3 temperatura | 6 oxigeno | | 1 temperatura |
| | ACTUADORES | | | |
| **ID TIPO** | 56 mensaje | 51 mensaje | 55 valvula | 53 valvula |
| **ID TIPO** | 59 valvula | 54 valvula | 57 mensaje | 52 mensaje |

En la tabla 7. podemos apreciar que los sensores y actuadores eliminados son aquellos que se encuentran en rojo. De esta manera, se deberían esperar resultados solo para los estanques 1 y 4 de tipo mensaje y para los estanques 2 y 3 de tipo válvula, se realizan las eliminaciones correctamente obteniendo los siguientes resultados:

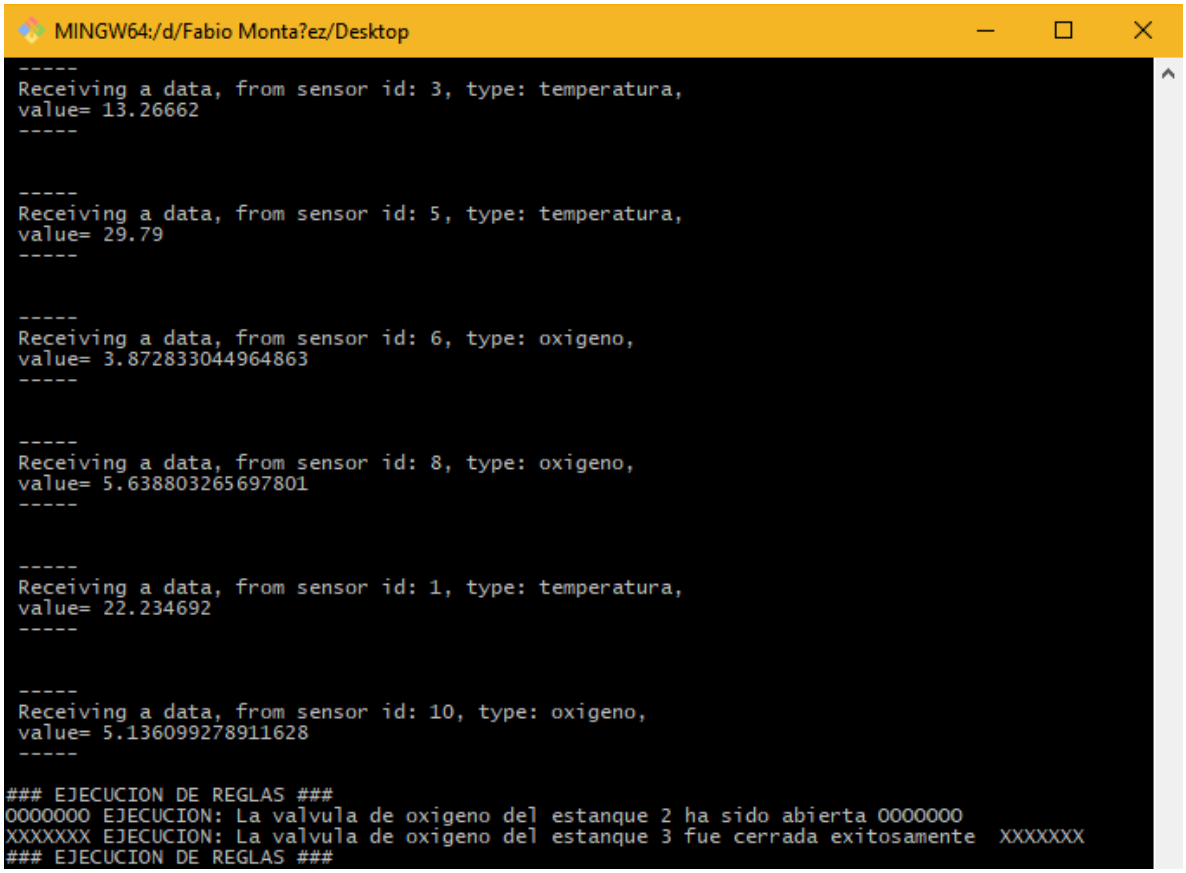
*Figura 25. resultados luego de la eliminación de dispositivos*




Como se puede evidenciar para los estanques 1 y 4 solo se emiten acciones de tipo mensaje, y para los estanques 2 y 3 solo se ejecutan acciones de tipo válvula.

Por último, eliminaremos estanques del contexto, situación que se puede presentar cuando un ciclo productivo ha terminado, o se desea desocupar un estanque para su mantenimiento, para este caso eliminaremos los estanques 1 y 4. Procedemos a eliminarlos y obtenemos los siguientes resultados:

*Figura 26. Eliminación de estanques del contexto*




Vemos como ya solo se están ejecutando acciones para los estanques 2 y 3 ya que en la plataforma se eliminaron los estanques 1 y 4 por lo que esta última prueba también tuvo el resultado esperado.

# 9. TRABAJO A FUTURO

- Implementar un framework ligero de inyección de dependencias para la plataforma que apoye la creación de las instancias de los sensores y actuadores esto con el fin de escribir menos código a la hora de implementar un nuevo dispositivo.

- Aunque el canal de comunicación entre el usuario y el “web services” es simple pero funcional, podría implementarse un formato de archivos más genérico (JSON en lugar de HTML) y la utilización de Javascript podría darle más dinamismo a los servicios WEB.

- A pesar de que la implementación en la placa Intel Galileo fue suficiente para la construcción del prototipo presentado, se recomienda utilizar sistemas embebidos con mayores recursos disponibles como lo son la placa Edison.

# 10. CONCLUSIONES

La piscicultura tradicional en Colombia necesita soluciones tecnológicas que apoyen los procesos para controlar su productividad. En este proyecto, se diseñó una plataforma software que ofrece la posibilidad de monitorear en tiempo real las características del ambiente en el proceso del cultivo de peces; este el primer paso en la contribución a la generación de soluciones a las necesidades en pro de la expansión de la piscicultura, ya que se basa en alternativas económicas, accesibles y prósperas debido al auge que se proyecta actualmente para las plataformas y arquitecturas embebidas.

El diseño presentado en este proyecto de grado da prioridad a dos características que son el corazón de la plataforma: la escalabilidad, que es uno de los valores agregados más importantes debido a que si la plataforma no permitiera la posibilidad de monitorear nuevos estanques que se sumen al ambiente piscícola, no hubiera tenido sentido llevar a cabo el proyecto, ya que la plataforma se pensó desde un inicio para que creciera de la misma manera que lo hace la piscicultura y no como un sistema cerrado, esto conlleva al segundo atributo destacable: una plataforma abierta, permitiendo que la plataforma se adapte rápidamente a nuevas necesidades emergentes evitando considerablemente que se vuelva obsoleta con el avanzar del tiempo.

La rápida implementación del prototipo que fue validado, demostró que las posibilidades para llevar a la realidad una plataforma estable, robusta y confiable a corto plazo son altas, estas características permitirán tomar decisiones más rápidas y acertadas ya que el control no estará basado en la experiencia si no en los datos reales del ambiente y al final esto se verá traducido en ciclos de producción más cortos, peces conviviendo en ambientes más saludables y productos más sanos para el consumo humano.

## REFERENCIAS


[1] Organización de las Naciones Unidas para la Alimentación y la Agricultura. Estado mundial de la pesca y la acuicultura (SOFIA) 2014. Disponible en: http://www.fao.org/3/a-i3720e.pdf

[2] Organización de las Naciones Unidas para la Alimentación y la Agricultura. Tecnología de la acuicultura. Disponible en: http://www.fao.org/fishery/technology/aquaculture/es

[3] Organización de las Naciones Unidas para la Alimentación y la Agricultura. FISHERY COUNTRY PROFILE: THE REPUBLIC OF COLOMBIA. Disponible en: ftp://ftp.fao.org/FI/DOCUMENT/fcp/en/FI_CP_CO.pdf

[4] Organización de las Naciones Unidas para la Alimentación y la Agricultura. National production. Sub-national and individual farm level data. Search by Country: Colombia (2009). Disponible en: http://www.fao.org/fishery/naso/maps/search/jsp/simpleSearch.jsp?country=CO&dataType=A&size=S

[5] Organización de las Naciones Unidas para la Alimentación y la Agricultura. Visión general del sector acuícola nacional: Colombia. Disponible en: http://www.fao.org/fishery/countrysector/naso_colombia/es#tcN7015F

[6] GOMEZ RONCANCIO, German. Diseño mecánico a bajo costo y construcción del modelo de una planta de tratamiento de agua para piscicultura ornamental. Bucaramanga: Universidad Industrial de Santander, 2007.

[7] TABARES, Wilson Enrique. Estudio de caso sobre la utilización de sistemas de generación fotovoltaicos en cultivos de piscicultura intensiva en el departamento de Santander. Bucaramanga: Universidad Industrial de Santander, 2016.

[8] PRADA CAMACHO, Wilson Ramiro. Diseño de un sistema embebido que permite hacer seguimiento y control a vehículos de transporte publico basado en BRT. Bucaramanga: Universidad Industrial de Santander, 2017.

[9] Fundación de la Innovación Bankinter. El Internet de las cosas. En un mundo conectado de objetos inteligentes. Disponible en: https://www.fundacionbankinter.org/documents/20183/42758/PDF+Internet+de+las+cosas/a94d3ba9-31da-4d43-a16a-a46ff99442d8

[10] Internet de las cosas: Objetos interconectados y dispositivos inteligentes (2013) Madrid. Clúster ICT – Audiovisual. Disponible en: https://actualidad.madridnetwork.org/imgArticulos/Documentos/635294387380363206.pdf

[11] AQUAHOY. Portal de Información en Acuicultura. Publicado el 9 de septiembre de 2015. Disponible en: http://www.aquahoy.com/noticias/peces/24835-fabricante-de-alimentadores-inteligentes-de-peces-recibe-financiamiento-de-aqua-spark-y-ideosource

[12] IoTHUB. How IoT is helping Tasmania's oyster industry. Publicado el 6 septiembre 2016. Disponible en: https://www.iothub.com.au/news/how-iot-is-helping-tasmanias-oyster-industry-436397

[13] Telefónica Business solutions. El IoT al rescate de la industria del marisco. Publicado el 22 septiembre de 2015.Disponible en: https://iot.telefonica.com/blog/el-iot-al-rescate-de-la-industria-del-marisco

[14] FIS Unión Europea. SK Telecom valida sistema de gestión IoT en granja de anguilas. Publicado el 2 de septiembre de 2014. Disponible en: http://www.fis.com/fis/worldnews/worldnews.asp?monthyear=9-2014&day=2&id=71037&l=s&country=0&special=&ndb=1&df=0

[15] Object Oriented Design. Factory Method Pattern. Disponible en: http://www.oodesign.com/factory-method-pattern.html

[16] Intel®. Galileo Datasheet. Disponible en: https://www.intel.com/content/dam/support/us/en/documents/galileo/sb/galileo_datasheet_329681_003.pdf

[17] Dfrobot. Waterproof DS18B20 Digital Temperature Sensor. Disponible en: https://www.dfrobot.com/wiki/index.php/Waterproof_DS18B20_Digital_Temperature_Sensor_(SKU:DFR0198)

[18] AtlasScientific. Dissolved Oxygen Kit. Disponible en: https://www.atlas-scientific.com/product_pages/kits/do_kit.html

[19] Dfrobot. PH meter. Disponible en: https://www.dfrobot.com/wiki/index.php/PH_meter(SKU:_SEN0161)#Documents

[20] Dfrobot. Turbidity sensor. Disponible en: https://www.dfrobot.com/wiki/index.php/Turbidity_sensor_SKU:_SEN0189

[21] Dfrobot. Analog ORP Meter. Disponible en: https://www.dfrobot.com/wiki/index.php/Analog_ORP_Meter(SKU:SEN0165)

[22] Dfrobot. Non-contact Liquid Level Sensor XKC-Y25-T12V. Disponible en: https://www.dfrobot.com/wiki/index.php/Non-contact_Liquid_Level_Sensor_XKC-Y25-T12V_SKU:_SEN0204

[23] Dfrobot. Relay Shield for Arduino V2.1. Disponible en: https://www.dfrobot.com/wiki/index.php/Relay_Shield_for_Arduino_V2.1_(SKU:DFR0144)

[24] Dfrobot. 12V Solenoid Valve 1/2". Disponible en: https://www.dfrobot.com/product-1530.html

[25] Adafruit. OLED Breakout Board - 16-bit Color 1.5" w/microSD holder. Disp en: https://www.adafruit.com/product/1431

[26] Dfrobot. RoMeo BLE. Disponible en: https://www.dfrobot.com/wiki/index.php/RoMeo_BLE_(SKU:DFR0305)

[27] Dfrobot. world-leading robotics and open source hardware provider. Disp en: https://www.dfrobot.com/

[28] Linux foundation collaborative projects. Yocto Project. Disponible en: https://www.yoctoproject.org/

[29] Google. Class EventBus. Disponible en: https://google.github.io/guava/releases/22.0/api/docs/com/google/common/eventbus/EventBus.html

[30] Easy Rules. The simple, stupid Java™ rules engine. Disponible en: http://www.easyrules.org/

[31] Eclipse. Jetty:// Disponible en:
http://www.eclipse.org/jetty/

[32] Intel® Galileo. Getting Started Guide. Disponible en:
http://download.intel.com/support/galileo/sb/galileo_gsg_329685008.pdf

[33] Intel® IoT. Getting Started with the Intel® Galileo Board on Windows. Disp en:
https://software.intel.com/es-es/get-started-galileo-windows

[34] GitHub. Repositorio del prototipo piscicultura en JAVA. Disponible en:
https://github.com/faanmogo/Piscicultura

# BIBLIOGRAFÍA

GOMEZ RONCANCIO, German. Diseño mecánico a bajo costo y construcción del modelo de una planta de tratamiento de agua para piscicultura ornamental. Bucaramanga: Universidad Industrial de Santander, 2007.

PRADA CAMACHO, Wilson Ramiro. Diseño de un sistema embebido que permite hacer seguimiento y control a vehículos de transporte publico basado en BRT. Bucaramanga: Universidad Industrial de Santander, 2017.

TABARES, Wilson Enrique. Estudio de caso sobre la utilización de sistemas de generación fotovoltaicos en cultivos de piscicultura intensiva en el departamento de Santander. Bucaramanga: Universidad Industrial de Santander, 2016.